AGARD-AR-325

AGARD

ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT
7 RUE ANCELLE, 92200 NEUILLY-SUR-SEINE, FRANCE

AGARD ADVISORY REPORT NO. 325

Knowledge-Based Guidance and Control Functions

(Application des systèmes experts pour le guidage et le pilotage)

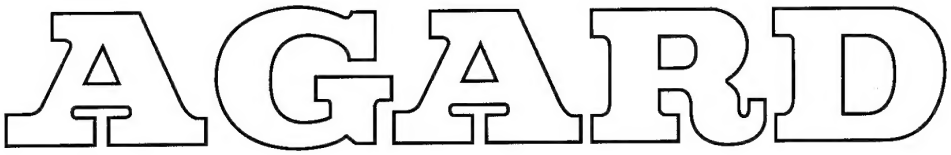
This report has been prepared as a summary of the deliberations of Working Group 11 of the Guidance and Control Panel of AGARD.

NORTH ATLANTIC TREATY ORGANIZATION

Published January 1995

Distribution and Availability on Back Cover

AGARD-AR-325



ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT

7 RUE ANCELLE, 92200 NEUILLY-SUR-SEINE, FRANCE

AGARD ADVISORY REPORT NO. 325

Knowledge-Based Guidance and Control Functions

(Application des systèmes experts pour le guidage et le pilotage)

This report has been prepared as a summary of the deliberations of Working Group 11 of the Guidance and Control Panel of AGARD.

North Atlantic Treaty Organization
Organisation du traité de l'Atlantique Nord

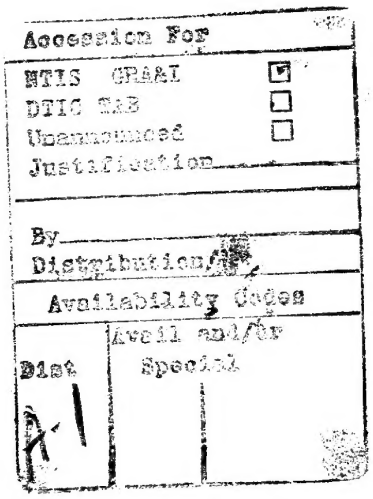
The Mission of AGARD

According to its Charter, the mission of AGARD is to bring together the leading personalities of the NATO nations in the fields of science and technology relating to aerospace for the following purposes:

- Recommending effective ways for the member nations to use their research and development capabilities for the common benefit of the NATO community;
- Providing scientific and technical advice and assistance to the Military Committee in the field of aerospace research and development (with particular regard to its military application);
- Continuously stimulating advances in the aerospace sciences relevant to strengthening the common defence posture;
- Improving the co-operation among member nations in aerospace research and development;
- Exchange of scientific and technical information;
- Providing assistance to member nations for the purpose of increasing their scientific and technical potential;
- Rendering scientific and technical assistance, as requested, to other NATO bodies and to member nations in connection with research and development problems in the aerospace field.

The highest authority within AGARD is the National Delegates Board consisting of officially appointed senior representatives from each member nation. The mission of AGARD is carried out through the Panels which are composed of experts appointed by the National Delegates, the Consultant and Exchange Programme and the Aerospace Applications Studies Programme. The results of AGARD work are reported to the member nations and the NATO Authorities through the AGARD series of publications of which this is one.

Participation in AGARD activities is by invitation only and is normally limited to citizens of the NATO nations.



Published January 1995

ISBN 92-836-1009-1

Printed by Canada Communication Group
45 Sacré-Cœur Blvd., Hull (Québec), Canada K1A 0S7

Preface

The AGARD Guidance and Control Panel has become increasingly more interested in applications of new concepts and tools arising from the field of investigation which has been termed "Artificial Intelligence". In particular, it was noted that Knowledge-Based Systems technology was maturing rapidly with guidance and control applications being discussed more frequently in Symposia papers and the literature in general. Working Group 11 on "Knowledge-Based Guidance and Control Functions" was established by the Guidance and Control Panel in the fall of 1990. The Terms of Reference approved by the National Delegates Board of AGARD were:

(1) Analyze the structure of knowledge-based guidance and control functions related to aircraft, missions, and the battlefield, and identify their potential for automation.

(2) Analyze the structure of knowledge-based guidance and control functions related to the life cycle of guidance and control systems, and identify their potential for automation.

(3) Review the state-of-the-art of those software- and hardware-oriented technologies required for the transfer of the knowledge-based G&C functions to automatic systems. In particular:

— Compare new and classical techniques and consider their optimal fusion.

— Consider the potential of neural networks.

— Consider the problems in safety-critical applications.

(4) Review existing programs.

(5) Make recommendations for future work.

The Working Group held six working sessions in conjunction with the Guidance and Control Panel Symposia of 1990-1993. This Final Report presents the material produced by the group. It is the result of a team effort and represents the consensus of the Working Group.

Préface

Le Panel AGARD du Guidage et du Pilotage s'intéresse de plus en plus à l'application des nouveaux concepts et des différents systèmes d'aide émanant du domaine de recherche appelé «intelligence artificielle». En particulier, il a été constaté que les technologies des systèmes à base de connaissances étaient en pleine évolution et que les applications guidage et pilotage figuraient de plus en plus fréquemment dans les communications de symposiums et dans la littérature en général.

Le groupe de travail n° 11 sur «Les fonctions du guidage et du pilotage à base de connaissances» a été créé par le Panel AGARD du guidage et du pilotage en automne 1990. Le mandat de ce groupe, tel qu'approuvé par le Conseil des délégués nationaux de l'AGARD, fut le suivant :

(1) analyser la structure des fonctions de guidage et de pilotage à base de connaissances liées aux aéronefs, aux missions, et au champ de bataille, et d'apprécier leurs possibilités d'automatisation;

(2) analyser la structure des fonctions de guidage et de pilotage à base de connaissances liées à la durée de vie des systèmes de guidage et de pilotage et apprécier leurs possibilités d'automatisation;

(3) faire le point sur les technologies de pointe en logiciel et matériel exigées pour le transfert des fonctions de guidage et de pilotage à base de connaissances aux systèmes automatisés, et, en particulier

— faire la comparaison des techniques nouvelles et classiques et étudier leur intégration optimale;

— étudier le potentiel des réseaux neuroniques;

— apprécier les problèmes qui sont à envisager dans les applications où la sécurité est un élément critique;

(4) évaluer les programmes existants;

(5) faire des recommandations concernant de futurs travaux.

Le groupe n° 11 a organisé six séances de travail conjointement avec les symposiums du Panel AGARD du guidage et du pilotage 1990-1993. Ce rapport final présente les textes élaborés. Il s'agit d'un véritable travail d'équipe qui traduit l'avis unanime du groupe de travail.

Members of the Working Group

Co-Chairman: Prof. Dr. rer. nat. Heinz Winter
Institut für Flugführung
Deutsche Forschungsanstalt für
Luft- und Raumfahrt
Braunschweig, Germany

Co-Chairman: Ray Van Hood
NASA
Washington, D.C.
United States

Members

Ing Winfried Büttner
Siemens AG
Unterschleissheim, Germany

Prof. Dr. -Ing. Karl Friedrich Kraiss
Lehrstuhl für Technische Informatik der
Rhein.-Westf. Technischen Hochschule Aachen
Aachen, Germany

Dr. Ing Uwe Voelckers
Deutsche Forschungsanstalt für Luft - und
Raumfahrt e.V.
Institut für Flugführung
Braunschweig, Germany

Professor Dr-Ing Reiner C. Onken
Universität der Bundeswehr München
Fakultät für Luft und Raumfahrttechnik
Neubiberg, Germany

Gilles Champigneux
Dassault Aviation
Saint-Cloud, France

Didier F. Godart
SAGEM
Cergy-Pontoise, France

Professeur Henry Kanoui
IRIAM
Marseille, France

Dr. Stéphane Sallé
SAGEM
Cergy-Pontoise, France

Jean-Michel Darroy
MATRA MARCONI-SPACE
Toulouse Cedex, France

Howard Howells
Defence Research Agency
Farnborough, Hampshire
United Kingdom

Ermanno Girardelli
Alenia Aeronautica DVD
Torino, Italy

Keith Rosenberg
GEC- Marconi Avionics Limited
Rochester, Kent United Kingdom

Dr. Milton B. Adams
Charles Stark Draper Laboratory
Cambridge, MA, United States

Harry A. Funk
Honeywell Technology Center
Minneapolis, MN, United States

Dr. Peter E. Friedland
Artificial Intelligence Research Branch
NASA Ames Research Center
Mountain View, CA, United States

Douglas I. Holmes
ISX Corporation
Westlake Village, CA, United States

Acknowledgements

The preparation of this report of the Working Group 11 would not have been possible without the help of many individuals and institutions whose efforts are highly appreciated. Leading experts in fields of activities related to the Working Group study have made contributions by presenting their ideas at the meetings, by writing Working Papers or by discussing their publications with the Working Group. Appendix 2 of this report describes the various contributions of these persons, together with a list of the papers and documents provided by them.

Special thanks are given to the Companies and Institutions which have made it possible for the Working Group to visit their laboratories to see and discuss successful applications of Knowledge-Based Systems. These visits have helped the members of the Working Group understand the state of the art and the application potential of this technology. Appendix 2 describes these visits, together with a list of the documents and papers handed out to the Working Group.

We also greatfully acknowledge the support of the parent Institutes and Companies of the Members of the Working Group for making available the time and effort of their staff.

Contents

List of Acronyms

ADAM	Aircraft Diagnostics and Maintenance
AFP	Automatic Flight Planner
AFWAL	USAF Wright Aeronautical Labs
AI	Artificial Intelligence
AP	Attack Planning
ART	Automated Reasoning Tool
ASPIO	Assistant for Single Pilot IFR Operation
ATC	Air Traffic Control
ATE	Automated Test Equipment
ATFM	Air Traffic Flow Management
ATM	Air Traffic Management
ATS	Air Traffic Services
BIT	Built-In Test
CAD	Computer-Aided Design
CASSY	Cockpit Assistant System
CATMAC	Cooperative Air Traffic Management Concept
CERMA	Centre d'Étude et de Recherche en Médecine Aérospatiale
CND	Can Not Duplicate
COMPAS	Computer-Oriented Metering Planning and Advisory System
DARPA	Defense Advanced Projects Agency
DM	Dialogue Manager
DOD	Department of Defense (U.S.)
DRAM	Dynamic Random Access Memory
DSS	Decision Support System
DWIM	Do What I Mean
EA	Execution Aid
EDIF	Electronic Design Interchange Format
EETO	Earliest Estimated Time Of Arrival
ETO	Estimated Time Of Arrival
EU	Expected Utility
FCMDS	Flight Control Maintenance Diagnostic System
FCS	Fire Control System (or Flight Control System)
FIM	Fault Isolation Manual
FLCS	Flight Control System
FOA	Focus of Attention
FR/ED	Fault Reporting/Expert Diagnostic system
G&C	Guidance and Control
GCP	Guidance and Control Panel
GCS	Guidance and Control System
HOTAS	Hands-On Throttle And Stick
ICAO	International Civil Aviation Organization
IFR	Instrument Flight Rules
INCO	Integrated Communications Officer
INS	Inertial Navigation System
JSC	Johnson Space Center
KB	Knowledge Base
KBS	Knowledge-Based System
KE	Knowledge Engineer
KOD	Knowledge-Oriented Design
KS	Knowledge Source
LRU	Line Replaceable Unit
MCC	Mission Control Center
ME	Monitor of Environment
MFS	Monitor of Flight Status
MMI	Man Machine Interface
MP	Mission Planner
MS	Monitor of Systems

MST	Mission Support Tool
MTBF	Mean Time Between Failure
MTBUR	Mean Time Between Unnecessary Removals
PA	Pilot's Associate
PE	Piloting Expert
PFA	Post Flight Analysis
PGG	Plan-Goal Graph
PIER	Pilot Intent and Error Recognition
PP	Path Panning
PVI	Pilot Vehicle Interface
RADC	Rome Air Development Center
RTDS	Real-Time Data System
RTOK	Re-Test OK
SA	Situation Assessment
SADT	Structured Analysis and Design Technique
SAHARA	Simulateur d'Architecture Hétérogène, d'Agent et de Ressource Active (Heterogeneous Architecture, Agent and Active Resource Simulator)
SCC	Spacecraft Control Center
SME	Subject Matter Expert
SS	System Status
STM	Standard Troubleshooting Methodology
TMA	Traffic Manager Advisor
TP	Tactics Planner
TRN	Terrain Referenced Navigation
UR	Unnecessary Removals
V & V	Verification and Validation
VME	Versamodule Europe
WG 11	Working Group 11

CHAPTER 1

INTRODUCTION AND BACKGROUND

1.1 Introduction and Background

Working Group 11 (WG 11) was established by the Guidance and Control Panel (GCP) in 1990 to evaluate Guidance and Control functions with respect to the knowledge-based content of actions carried out by human operators. Such functions are performed, for example, in the cockpit of a military or civilian aircraft, at an air traffic controller's work position, at a mission planning work station or in a space vehicle control center. The Terms of Reference of WG 11 contained the following objectives:

- Analysis of the G&C functions related to aircraft, mission and the battlefield, and their potential for automation. This analysis shall also include the management of the life cycle of G&C systems
- Review of the technologies available for the automation of these functions.
- Review of existing programs in the NATO countries, and recommendations for further work.

WG 11 had 6 Meetings between Fall 1990 and Spring 1993, and made site visits to industrial and research facilities in Germany, France and the United States (see Appendix 2).

We begin the discussions of knowledge-based functions by considering the general structure of human behavior. The goal-directed interactions of **man** with the surrounding **world** can be decomposed into the functional elements (subfunctions) of the **recognize-act-cycle** (or stimulus-response-cycle) [1.1, 1.2] :

1.) Recognize the actual state of the world and compare it with the desired state (which corresponds to the goal of the interaction). (MONITORING)

2.) Analyse the deviations of actual and desired state. (DIAGNOSIS)

3.) Think about actions to modify the state of the world. (PLAN GENERATION)

4.) Decide the necessary actions to reach the desired state. (PLAN SELECTION)

5.) Take the necessary actions to change the state of the world. (PLAN EXECUTION)

For many simple tasks a person's physical sensors (eyes, ears, etc.), his brain and his physical effectors (arms, legs, etc.) are sufficient to carry out these functions (manual interaction). More demanding tasks (e.g. flying a military airplane) go beyond the capabilities of his physical sensor/effector equipment. Therefore, man has invented a great variety of **tools** to support his interactions with the world. As shown in Figure 1.1, the tools may support or even replace the human functions.

Three types of tools may be distinguished:

- **sensor systems** to support functional element 1.)
- **information processing systems** to support functional elements 2.) , 3.) and 4.), and
- **effector systems** to support functional element 5.).

In our industrial society many of the human interactions with the world occur in a **work system** [1.3, 1.4, 1.5], where man spends a great part of his life. A system is an arrangement of elements whose relationships serve a certain purpose or goal. The goal of a work system is to fulfill a certain **task**, for which the work system has been built. Figure 1.2 illustrates the generic structure of such a work system.

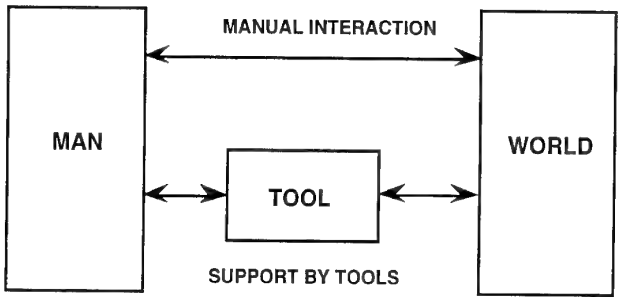


Figure 1.1 Generic Structure of the Interaction of Man With the World

This Work System structure consists of the following elements: Operator, Work Object and the Tool(s). The tools are devices or machines (or sometimes other people) which help the operator to fulfill the task. The work objects are materials, goods, data and (in some work systems) also people which have to be modified, used, transported etc., corresponding to the task. The system elements interact through the operator- and the work-object-interfaces, with the goal to produce a certain output, the product. The operator can interact with the work-object directly (manual operation) or with the help of the tool (semi-automatic or automatic operation). The **declarative (or object-oriented) representation** which describes the elements making up the work-system in Figure 1.2 is instantiated in that figure with the situation of a cockpit in an airplane. Here the operator is the pilot, the work-object is the airplane and the tool is the Guidance and Control System (GCS) of the aircraft. The goal is to fly the airplane in accordance with the flight plan (or the mission plan in the military case) subject to the ground rules of safe flight and possible air traffic control (ATC) directives. The functional relationships of the elements of the work system will be discussed below in more detail.

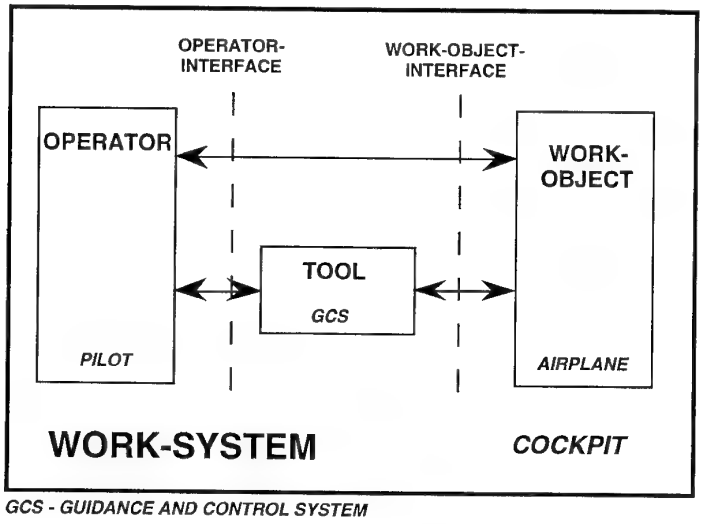


Figure 1.2 Generic Structure of a Work System

A framework has been described by J. Rasmussen [1.6], which can be used to provide a better understanding of the interactions of a human operator with a tool or a work-object. Following his ideas, these interactions can take place on three levels:

- **Skill-based activities** which are carried out almost subconsciously and automatically. A skilled operator has a large repertoire of automated sensory-motor subroutines which he can put together to form larger patterns of behavior in order to interact with the tool or the work-object.
- **Rule-based activities** which are steered by stored rules that have been learned during instruction or by experience. These rules cover all routine situations for which there are no automated sensory-motor subroutines.
- **Knowledge-based activities** which take place in non-routine situations, where no learned rules or skills can solve the problem. In such situations the operator has to develop new problem solutions based on his objectives and knowledge about the work-object and the world.

The following Table 1.1 describes some of the characteristics of these interaction levels.

Table 1.1 Characteristics of the Levels of Interaction Between Human and Work-object

	Required Response Time	*Structure of Information Processing*
Skills	short	connectionist
Rules	medium	cognitive
Knowledge	long	cognitive

By analogy to models of cognitive science one can assume that skills are stored in the human brain as "situation/action" patterns, rules in the form of symbolic "if(situation)-then(action)" pairs. Knowledge can be represented in declarative form (knowledge about the world, the work-object, the tools and the human operator) or in procedural form (knowledge about actions and their use for problem solving).

The operator of a G&C system performs his information processing functions as part of the recognize-act-cycle:

- By visually scanning the displays, he continuously monitors the state of the aircraft and mission environment under his control.
- He processes the obtained information in order to derive any necessary corrective actions in the case of a deviation between the actual and the desired state.
- He implements the necessary actions by direct interaction with the G&C system.

In the most general form, information processing functions carried out by human operators can be decomposed into the following subfunctions:

- Recognition of a problem in connection with the actual state of the work-object and its representation in a "mental model". Definition of the desired goal state.
- Construction of control operations to bring the work-object from the recognized problem state to desired goal states.
- Selection of criteria to evaluate the different control strategies.
- "Simulation" of the effect of the control strategies on the work-object.
- Evaluation of the possible control strategies.
- Selection of the appropriate control strategy to "best" drive the work-object to the desired goal state.

An **experienced operator** has gone through this scheme many times and has been trained to recognize typical problems and to react to them in an appropriate manner. Based on what he has learned (gained experience) from these repetitive familiar situations, he immediately has a mental model available for the problem representation, and he also knows the criteria to be applied. He knows which control strategies are available to solve the problem. He

also knows their effect on the work-object and their relative value with respect to the criteria. In analogy to models of cognitive science one can assume that this experience is stored in his long term memory in the form of **rules**. For this experienced operator the information processing reduces to the simpler form:

- Selection of possible rules to drive the work-object to the goal state.
- Evaluation of these rules and selection of the appropriate control strategy.

After many years of experience, an **expert operator** has learned the appropriate control strategy for all routine problem situations. His information processing reduces to a kind of "pattern matching" process:

- For each routine problem of G&C he has the appropriate control strategy immediately at hand. This experience is stored in his Long Term Memory as a collection of "situation/action" patterns.

These patterns are activated by the information obtained from his sensory inputs. It is not difficult to see that these three models for the information processing of an operator are related to the three levels of interaction (knowledge, rules and skills) discussed above.

Figure 1.3 shows how the work-system is embedded in the environment: It receives an **input** in the form of material, energy, information (or a combination of the three), from which it produces an **output** (also as material, energy or information), in the form of the product for which the work-system was built. The general scheme is again instantiated in this figure with the example of a cockpit in an airplane. The operator is the pilot, his tool is the GCS of the aircraft and the work-object is the airplane. The goal of this work-system is a safe flight corresponding to the flight plan (or the mission plan in the military case). The flight plan is the **main input** into the work-system. It is certainly an information input, and the pilot's (or the tool's) commands to the aircraft (corresponding to the flight or mission plan) are information outputs.

Such work-systems are the objects of research in Ergonomics and Human Factors Sciences [1.3, 1.4, 1.5]. In the general case, side-inputs and the impact of the environment have to be considered in addition to the main input. The system normally produces certain by-products and waste in addition to the intended main product, and also has an impact on the environment.

Another (complementary) way of viewing a G&C system as a work-system is the **procedural (or function-oriented) representation** in Figure 1.4, which describes those functions performed by the system elements that are required in order to obtain the product. It shows the inputs to the **work process** as:

- Information, in the form of a flight plan and
- information about the **state vector** of the airplane.

In manual flight, the pilot controls the aircraft in the recognize-act-cycle discussed above. So we can describe the **top level G&C functions** as

- Monitoring
- Diagnosis
- Plan generation
- Plan selection
- Plan execution.

In performing these top level G&C functions the pilot transform the aircraft state into its desired value, feeding the output of the work-process (the control commands) to the effectors (the actuators of the airplane). In the case of a semi-automatic or automatic flight, G&C tools contribute to performing (partially or totally) the top level G&C functions.

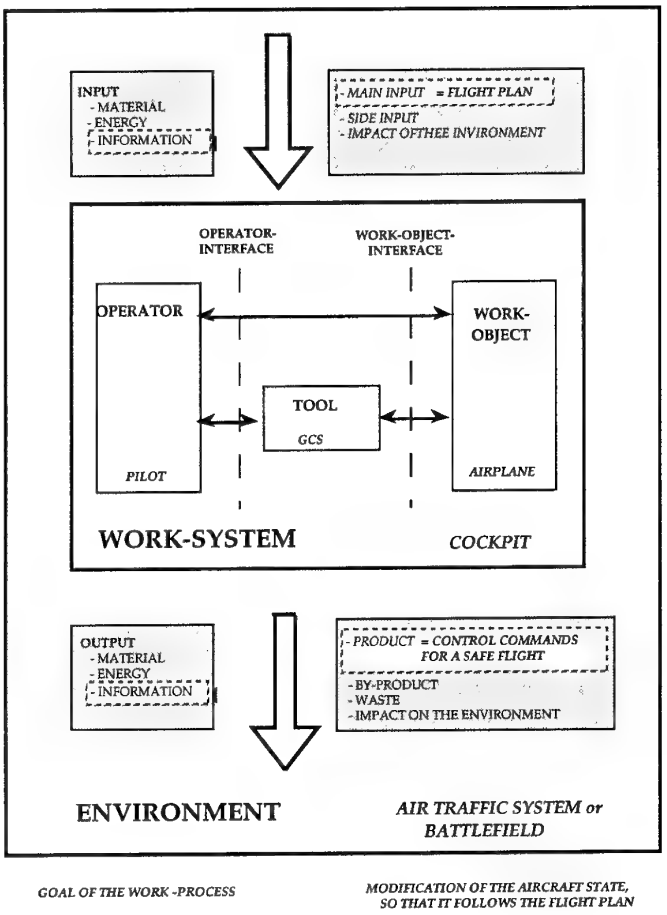


Figure 1.3 G&C of an Aircraft as a Work-System *(Declaritive Representation)*

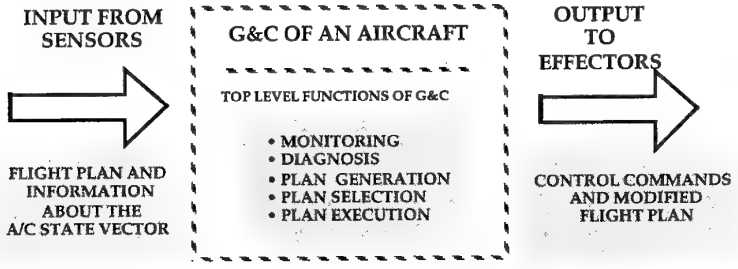


Figure 1.4 G&C of an Aircraft as a Work Process (Procedural Representation)

Complex tools - like those to be used for G&C - have a **life-cycle**. Corresponding to their use in a work-process, they have to fulfill certain requirements. The phases of their life-cycle are:

1. Requirements analysis
2. Design
3. Implementation
4. Installation and test
5. Use
6. Field support and

(7. Removal)

Work-objects - like an airplane - have the same life-cycle as described for the tools. Figure 1.5 shows in more detail the life-cycle phases **use** and **field support**. These are the phases where the tool (G&C system) and the work-object (airplane) are parts of the same work-system, as shown in the Figure 1.6.

During their life-cycle the tools (and work-objects) themselves go through several work-processes (corresponding to the different phases of the life-cycle). Similar knowledge is used by the operators in these work-processes. We introduce the term **life-cycle management** to describe a new function which organizes the transfer of this knowledge from one work-process to the other throughout the whole life-cycle.

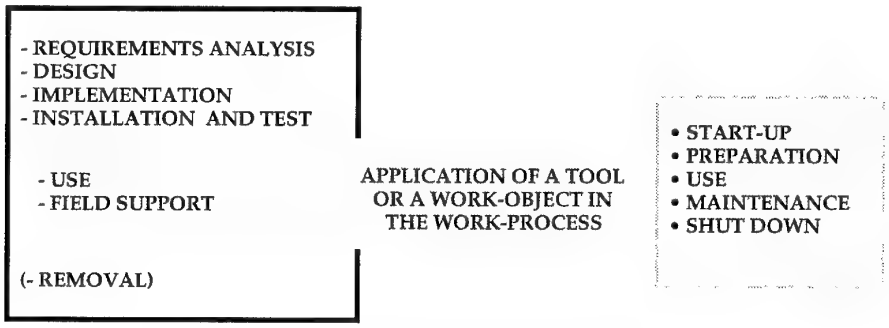


Figure 1.5 Life-Cycle of a Work-Object or a Tool

In the following chapters of this report the G&C subfunctions are considered in more detail in the domains of

- aircraft flight
- aircraft mission control
- air traffic control
- space vehicle control and
- life cycle management.

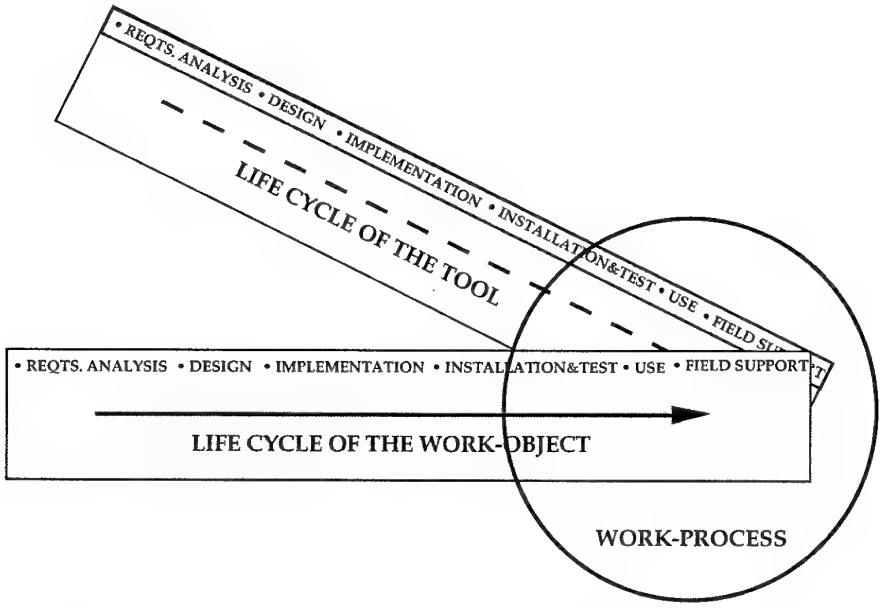


Figure 1.6 Work-Process and Life Cycle of Tool and Work-Object

They are analysed with respect to the possibility of building tools for the support of the knowledge based functions of human operators. Special attention is therefore payed to the **enabling technologies**. These are software and hardware technologies which are needed to build automated tools able to perform the G&C subfunctions . For the information processing functions in G&C the following technologies are relevant:

Computer hardware:	->	von Neumann computers
	->	highly parallel computers.
Computer software	->	numerical programming
	->	symbolic programming
	->	neural networks.

Von Neumann computers and numerical programming techniques are considered here as "traditional" technologies, the remaining ones as the "new" technologies. The traditional and new technologies are used to build the tools to support humans in carrying out the G&C subfunctions. Usually this support consists of a duplication of parts of the functions carried out by the operator and now also performed in parallel by the tools. Developments in G&C show that this "mixed mode" of operation is the most important one to be considered here. In this report, the new technologies are of particular interest, because many of the considered G&C subfunctions require the explicit usage of knowledge for their performance. So the potential of the new technologies for the implementation of knowledge into machine tools has to be discussed in detail. Worked-out examples are presented to exemplify the value of the new technologies for G&C automation.

The results of the functional analysis in G&C related domains is presented in the Chapter 2 of the report. The analysis contains the areas of flight control, navigation, mission management, air traffic management, space mission management, and life cycle management. The Chapter 3 discusses human factors aspects in relation to automation and the sharing of knowledge-based functions between humans and automatic systems. Available technologies for the automation of knowledge-based functions are discussed in the Chapter 4. The Chapter 5 is devoted to a description of the existing supporting infrastructures and development environments for knowledge-based systems. In Chapter 6, examples of current applications of automated knowledge-based functions are presented. Future trends and recommendations for further work are discussed in the Chapter 7. Appendix A contains a description of the potential of Neural Networks for the automation of knowledge-based functions. Appendix B describes the site visits of the WG, and the written material which was presented to and discussed by the WG.

1.2 References

[1.1] K.R. Boff, L. Kaufmann and J.P. Thomas (Eds.): *Handbook of Perception and Human Performance.* Vol. I and II. John Wiley and Sons, New York 1986.

[1.2] B.H. Kantowitz and R.D. Sorkin: *Human Factors: Understanding People-System Relationships.* John Wiley and Sons, New York 1983.

[1.3] H. Luczak *Arbeitswissenschaft.* Springer Verlag, Berlin, 1993.

[1.4] *Methodenlehre des Arbeitsstudiums.* REFA, Verband fÅr Arbeitsstudien und Betriebsorganisation e.V. Hanser Verlag, MÅnchen 1984.

[1.5] R. Bokranz und K. Landau: *Einführung in die Arbeitswissenschaft.* UTB 1619, Ulmer Verlag, Stuttgart 1991.

[1.6] J. Rassmussen: *Skills, rules and knowledge.* IEEE Transactions on Systems, Man, and Cybernetics. Vol. SMC-13, No. 3, May/June 1983.

CHAPTER 2

FUNCTIONAL ANALYSIS OF GUIDANCE AND CONTROL RELATED DOMAINS

2.1 Top Level Functional Structure of Guidance and Control Work Processes

Performing a functional analysis is the first step in a methodical approach to developing a solution to a complex problem. Indeed, understanding the basic functionality required of a solution is a prerequisite to the formulation of a design and ultimately the implementation of that design as the embodiment of the solution. In this section, a top level functional analysis of a real-time, closed-loop work process[1] is formulated. The analysis is performed at a high level of abstraction in order to insure that it captures the basic functionality of a broad class of Guidance and Control related work processes including those which implement decision-making, problem-solving and planning functions. Although not discussed explicitly in this section, the technical distinctions that can be drawn in differentiating among *planning, decision-making* and *problem-solving* are addressed in Chapter 3.

The functional analysis of a generic complex, real-time, closed-loop work process presented in this section serves to establish a common framework and a common language for describing a variety of specific guidance and control related work processes in ensuing sections. The two principal components of a functional analysis performed for any system are (1) a functional decomposition and (2) a detailed description of the information flow contained in the interfaces between the individual functions comprising that decomposition. The approach taken here is the classical structured analysis with a dataflow representation. A dataflow representation has been chosen because it readily lends itself to a hierarchical description. That is, each individual function can be further decomposed into more refined subfunctions which, when taken together, have combined inputs and outputs which are consistent with those of the original function. Neither control flow representations nor the emerging object-oriented analysis are addressed.

The functional analysis of a complex, real-time, closed loop work process given here is not claimed to represent the uniquely "correct" decomposition. Indeed, what is presented here reflects similarities with descriptions given elsewhere for real-time planning and decision-making systems, [e.g., 2.1, 2.2]. Our intent is to develop a description at a sufficiently high, abstract level that can be used as a template for the functional description of a variety of work processes ranging from air traffic flow control or the hierarchy of decision-making, problem-solving and planning functions required in a battlefield setting to the onboard functions required for the planning and execution of a mission of a tactical fighter aircraft.

Ultimately, the objective of the functional analysis is to identify that subset of functions within a real-time, closed loop work process which might be designed and implemented via knowledge-based approaches. The analysis in this section establishes a common framework and serves as a point of departure for the more detailed functional descriptions that follow in succeeding sections.

This section defines work processes and their internal functionality in a manner that is independent of the mechanisms that ultimately will be used for their implementation: i.e., man vs. machine, hardware vs. software. Allocation of function to person or machine or to a person and machine in concert is addressed in Chapter 3. Indeed, in so-called "human centered" designs [2.1-3] or "mixed-initiative" systems [2.1-4], the extent of human participation in each function is dynamic and can range from totally manual to completely autonomous.

2.1.1 System Functional Decomposition: Generic

Our presentation of the functional analysis of a generic closed-loop decision making system begins with a description of the basic elements depicted in Figure 2.1. A more detailed functional breakdown follows the generic system description and includes an overview of each of the decision-making functions contained in that more detailed analysis.

---

1 Here, the term work process has the distinct connotation as described in Chapter 1.

2.1.1.1 Basic Work Process Sub-Functions Comprising the Generic Decomposition

The generic decomposition of a real-time, closed-loop work process illustrated in Figure 2.1 comprises a basic set of individual sub-functions including: *Coordination*, *Situation Assessment*, *Plan Generation*, and *Plan Implementation*. Coordination translates external inputs into problem-solving strategies, objectives and constraints that are to be employed by the other sub-functions. The Plan Generation and Plan Implementation functions do exactly as their names suggest. Situation Assessment monitors the state of the world to determine whether the objectives and constraints embodied in the current operational plan/solution[2] are being honored as well as to detect previously unforeseen opportunities that may allow the accomplishment of more goals or objectives than those embodied in the current plan. The result of this monitoring can be either a decision to continue to pursue the current plan or to create a new plan in order to accommodate problems or to take advantage of opportunities. Note again that each of these functions is influenced by the output of the Coordination function as well as signals/data/information that are internal to the Assess-Plan-Implement loop shown in Figure 2.1.

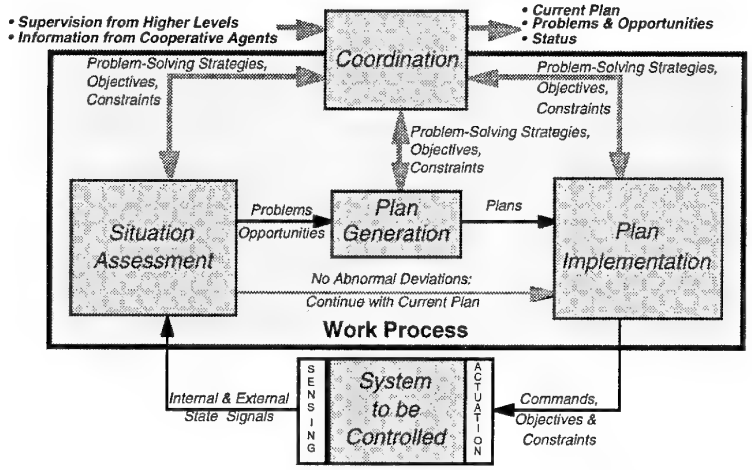


Figure 2.1 Functional Decomposition: Basic Sub-Functions of a Generic Work Process

The decomposition of the generic work process illustrated in Figure 2.1 provides a framework for the more detailed discussions that follow. Indeed, within this framework one is capable of describing the functionality of closed-loop systems ranging from classical closed-loop control systems to real-time hierarchical problem-solving such as battlefield management. Before, addressing the application of this analysis to those types of problems, we will expand slightly on this initial view of the generic functional analysis.

2.1.1.2 Inputs and Outputs of a Work Process Sub-Function

In the process of formulating a plan/decision/solution, a work process sub-function requires several classes of input information. These classes (shown entering from the top left in Figure 2.2) include problem-specific, real-time data and signals describing the elements of the current state of the world that are relevant to the task of the sub-function. In addition, tasking and strategy *control* inputs are required to define problem-solving criteria such as time available to generate a solution, the overall objective function, costs and constraints. These control inputs are influenced by inputs from a higher level entity and are shown entering at the top of the figure. Finally, in order to generate a solution, *knowledge* inputs including quasi-static, a priori information regarding the state of the world such as maps as well as problem-solving mechanisms that may be employed e.g., heuristics, search algorithms and inferencing techniques must be known. The knowledge inputs are shown entering at the bottom of the figure. The standard format used in pictorially depicting work processes in the ensuing sections shows the *control* inputs with gray arrows but does not explicitly show the *knowledge* inputs. This does not imply that *knowledge* inputs are considered to be unimportant nor that they should be overlooked, rather we have chosen only to represent explicitly the dataflow for the more dynamic, real-time information.

---

2 The terms *plan* and *solution* are often used interchangeably throughout this section.

The outputs representing the results of the decision-making can potentially include any or all of the classes of data and information that have been described above as inputs. These outputs, of course, serve as inputs to other sub-functions of the overall closed-loop work process.

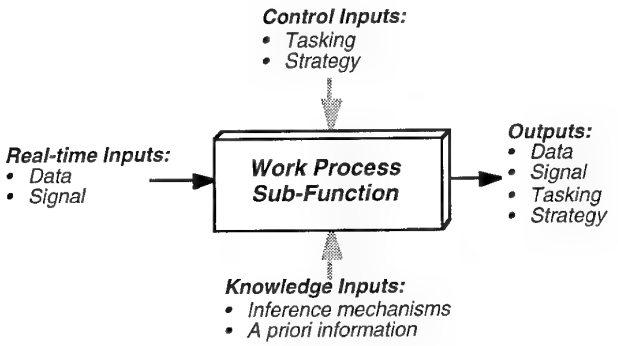


Figure 2.2 Generic Sub-Function

2.1.1.3 More Detailed Functional Decomposition

In order to more clearly describe the functions that are executed within the work process during normal operation, the Situation Assessment and Plan Implementation functions have been further decomposed into Diagnosis and Monitoring and Plan Selection and Plan Execution, respectively. This further decomposition makes it explicit that Diagnosis, Plan Generation and Plan Selection are required within the real-time loop only when Monitoring detects a significant deviation from the expected situation. This, along with a more detailed view of the Coordination function, is shown in Figure 2.3. In particular, the Coordination function is further decomposed to show both that it must organize and execute communications, when appropriate, with *external* agents and that it must continuously provide *internal* coordination to influence and control the execution of the sub-functions within the closed-loop work process.

2.1.1.4 Description of each Sub-Function

The discussion of a generic sub-function presented earlier in Section 2.1.1.2 defines the classes of inputs and outputs for each function and thereby serves as a basis for describing the inputs and outputs of the specific, individual work process sub-functions shown in Figure 2.3. The discussions below focus on information and the signals flowing within the real-time, decision-making loop. The description of each individual sub-function is, in some cases, contained implicitly in that of its inputs and outputs, i.e., the function transforms its inputs into its outputs. The mechanisms ultimately realize these transformations are not suggested here as they are the subject for later discussions in the context of specific G&C problems.

2.1.1.4.1 Situation Assessment

Monitoring

Inputs: Signals containing information about both the internal and external states of the system, i.e., the internal health and status of the *system-to-be-controlled* and the elements of the state of the environment external to the system-to-be-controlled that may influence the decision-making process, are provided as inputs from sensing systems. Problem-solving strategies that are part of the input include measures for and characterization of what constitutes a "significant" deviation from the expected state of the system. These are required to determine whether an extension of the normal operating loop to include Diagnosis, Plan Generation and Plan Selection is required.

Outputs: The output of the monitoring function is its decision indicating that either there are no abnormal deviations or there are significant deviations from expectations. In the former case, control is passed to the Plan Execution function. In the latter case, control is passed to the Diagnosis function along with an indication of the nature of the deviations. Although not shown explicitly in Figure 2.3, the state estimates (and uncertainties in those estimates) employed and/or generated by the Monitoring function are made globally available to all of the individual work process sub-functions.

Diagnosis

Inputs: The principal input is the output of the monitoring function: an indication of a deviation from expectations.

Output: In the case that a problem has been detected by the monitoring function, the output of Diagnosis is a determination of the source of the problem and a characterization of the impact of that problem on the capabilities of the system-to-be-controlled and/or on the ability of the system to pursue the current plan (e.g., an inability to achieve current or future objectives contained in the plan). In the case that the deviation from expectations represents a potential opportunity, the diagnosis function must characterize that opportunity in terms that will allow the Plan Generation function to attempt to incorporate any attendant new goals or objectives into the plan of activities to be pursued. Diagnostic information, similar to state estimates produced by the Monitoring function, are made globally available.

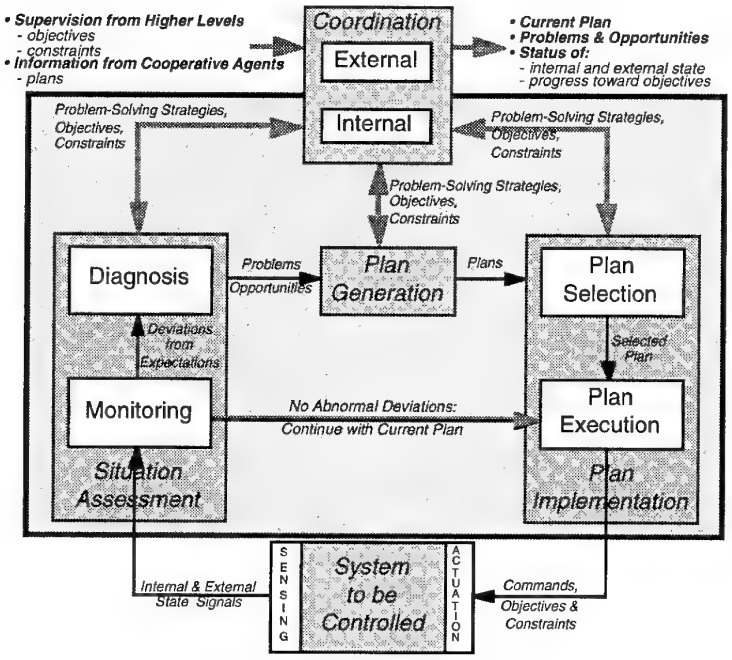


Figure 2.3 Functional Decomposition: Detailed Generic View

2.1.1.4.2 Plan Generation

Inputs: A new plan may be required in response to either detected and diagnosed problems or unexpected opportunities that are provided as input from the Diagnosis function. The Plan Generation function may generate a variety of plans that trade off among different levels of constraint and/or different objective functions (i.e., plan optimization criteria). The objective function(s) and constraints are provided by the Coordination function. Furthermore, a variety of algorithms or search techniques may be applied in generating plans. These mechanisms for plan generation are knowledge inputs. The decision as to which plan generation mechanism(s) to employ is made as a function of the strategy component of the control inputs from the Coordination function. For example, a quick heuristic may be required if a new plan is needed immediately to accommodate a serious (potentially mission or safety critical) problem that has been diagnosed. In other situations, it may be acceptable to continue to pursue the current plan while a more considered search of the plan space is executed in an attempt to refine the current solution to include additional opportunities or to accommodate minor degradations in the capabilities of the system-to-be-controlled.

Outputs: The output is the plan or set of plans that has been generated along with the figures of merit for (i.e., the values of) those plans and the resources required for their execution.

2.1.1.4.3 Plan Implementation

Plan Selection

Inputs: Given the plans that have been generated by the Plan Generation function, the Plan Selection function must choose from among those plans the one that best achieves the overall objectives as provided by the coordination function. For instance, there maybe multiple objective functions, and given a strategy for selecting among a set of plans, the Plan Selection function must make the choice of a single plan.

Indeed, a change in plan may not be warranted if there is no plan in the set of generated plans whose value sufficiently exceeds the value of the current plan[3]. The interpretation of "sufficiently exceeds" is made in the context of strategy inputs from the coordination function.

Output: The selected plan is the output and it is made globally available to all sub-functions.

Plan Execution

Input: Control is passed to Plan Execution either from Plan Selection, in the case when a new plan is generated and selected, or from Monitoring, in the case when no abnormal deviations are detected. In the former case, the current plan is updated by splicing the new plan onto the current plan at a designated point in time (either immediately or at some point in the future). The "splice time-point" and associated expected state of the system at the splice time are contained as elements of the new plan. Given the current state of the system, Plan execution interprets the current (or updated current) plan and creates the commands for the system-to-be-controlled. Execution of these "set-point" commands by the system-to-be-controlled results in the pursuit of the current plan.

Note that even when there are no detected abnormal deviations, under normal operations there typically will be minor deviations from the expected state. Thus, in addition to creating set-point commands for the pursuit of the current plan, an auxiliary role of the Plan Execution function is to determine "perturbation" commands that will correct for the range of normally expected deviations of the actual system state from the planned state.

Outputs: Commands in the form of both objectives and constraints for the system-to-be-controlled are the output of the Plan Execution function.

2.1.1.4.4 Coordination

The performance of the Internal and External Coordination functions are key to the overall performance of the work process. Concomitantly, their functional analysis and design are probably the most challenging (and, unfortunately, the most often neglected or overlooked). If allocated to a machine (i.e., if automated), internal and external coordination represent a significant potential for solution by and challenge for Knowledge Based technologies.

External Coordination

The External Coordination sub-function is responsible for receiving and analyzing inputs from external agents: those external agents may be either higher level supervisory agents or cooperative agents solving problems or making decisions at the same level. External Coordination is also responsible for assembling and transmitting information to those same agents. Here "assembling" implies deciding (1) what to transmit, (2) when to transmit it and (3) to whom to transmit.

Examples of interactions with external agents:

- Decide what gathered and assessed information to communicate
- Determine what decisions/plans to communicate
- Request assistance from supporting agents
- Interpret requirements from higher planning authorities
- Negotiate with subordinate and collateral planning levels

---

3 A re-evaluation of the current plan is required in the face of any diagnosed problems with either the system to be controlled or the external environment within which that system must operate

Inputs:

Inputs include supervisory commands or information from higher levels in the form of objectives and constraints as well as plans or solutions generated by other agents at the same level of problem solving. In addition, information regarding elements of the state of the world that are relevant to the problem to be solved are also received.

Outputs:

The outputs include the current plan or solution, the status of the internal and external states and progress toward the accomplishment of the objectives in the current plan or solution. In addition, any diagnosed problems or opportunities that may be of interest to external agents are also included as outputs.

Internal Coordination

The principal function of Internal Coordination is to develop criteria and strategies for controlling or guiding the real-time decision-making performed by other functions within the work process. In particular, (1) by monitoring the assessed situation including progress toward the current solution and the state of both the system-to-be-controlled and the external environment and (2) by taking into consideration any plans developed by other agents and objectives and constraints input from higher level authorities, Internal Coordination develops strategies for controlling the other individual work process sub-functions including providing the criteria for deciding when replanning is required, the time allocated to generating a solution and cost/objective functions and constraints to be employed in generating a solution.

2.1.2 Functional Decomposition: Aircraft G&C Work Process

In order to make the description of the elements of the generic functional decomposition given in Section 2.1.1 more concrete, that description is elaborated in the context of an abstraction of an aircraft G&C work process. The inputs and outputs of the individual functions contained in the decomposition are illustrated in Figure 2.4 and are presented in outline form in Table 2.1. The aircraft G&C work process as well as other related problems are addressed in further detail in later sections of this report.

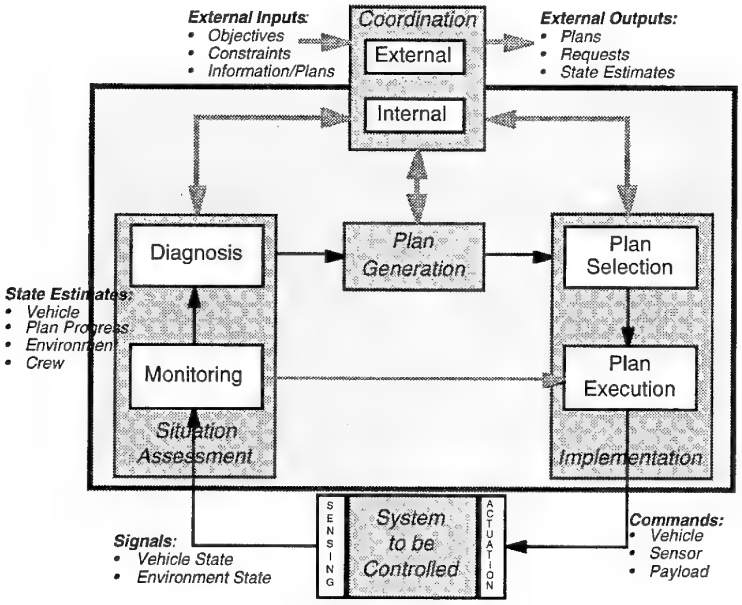


Figure 2.4 Functional Decomposition of a Knowledge-Based G&C System

Table 2.1 Aircraft G&C System: Functions, Inputs and Outputs

Function	*Control Inputs*	*Knowledge Inputs*	*Data/Signal Inputs*	*Output*
Monitoring	- Thresholds for determining what constitutes an abnormality	- A Priori information - Estimation algorithms	- Internal system sensors, e.g.,: subsystem health and status navigation - External system sensors, e.g.,: weather threats terrain	- Estimates of both internal vehicle and crew states and external mission environment states relative to planned/predicted states - Uncertainty in (quality of) estimates
Diagnosis		- Subsystem models - World models	- Estimates of, e.g.,: health and status navigation weather threats terrain	- Source/cause of detected vehicle or payload problems or changes in mission environment - Characterization of potential mission opportunities - Inferred missing data - Prediction / anticipation of significant future mission events
Plan Generation	- Objectives - Constraints - Replanning urgency	- Planning Algorithms, e.g.,: trajectory sensor management payload	- Problems - Opportunities	- Plans
Plan Selection			- Current Plan - New Plans	
Plan Execution		- Mechanisms for translating plans into commands for the system-to-be-controlled, e.g., : trajectory sensor payload	- Current and newly selected plans - Vehicle and mission states expected states estimated states	- Mission, trajectory and information gathering commands - Constraints trajectory sensor payload
Internal Coordination			- Monitor the assessed situation: Progress toward current plan State of the vehicle State of the environment	- Decide when to replan - Develop inputs for planning functions: strategies objectives constraints
External Coordination				- Decide what gathered and assessed information to communicate - Determine what decisions/plans to communicate - Request assistance from supporting agents - Interpret requirements from higher planning authorities - Negotiate with subordinate and collateral planning levels

2.1.3 Hierarchical Decomposition

In order to make the solution of complex problems tractable, they are often decomposed into simpler, decoupled subproblems that can be solved (nearly) independently [2.5 - 2.10]. If the decomposition is formulated with proper coordination[4] of the processes generating the solutions to the subproblems, then the set of solutions for the subproblems can be combined into a near-optimal, complete solution for the original, more complex problem. The coordination of the solutions to the subproblems is managed by a *Master Problem Solving Level.*

---

4 Note that here we use the terms *coordination* and *subproblems* in a different context than we had earlier in the discussions of internal and external coordination.

The selection of the individual subproblems to be addressed at each level, the specification of their local performance criteria and the modeling of their interactions are all part of an integrated approach to developing a hierarchical decomposition. An important goal in these selections is to maintain a balance in the complexity of decision-making effort across all levels of the hierarchy. The following basic questions must be answered when designing a hierarchy:

- How many levels are required?
- How should problem-solving be partitioned across levels?
- What constraints and objectives should be passed from level to level?
- What happens when a level cannot meet its objectives and/or honor its constraints?

Analytical approaches to decompositions of *Large Scale Optimization Problems* and the essential role played by the subproblem coordination function (which is itself formulated as an optimization problem) have been developed over the last three decades [2.5, 2.6]. These approaches have been extended in the development of methodologies for the decomposition of *Large Scale Control Problems* [2.7, 2.8]. These formal analytical developments help to establish methodologies for achieving decompositions whose subproblems are properly coordinated via a higher, Master, level.

Two types of problems which are amenable to hierarchical decompositions are described in the following: (1) Spatially and functionally distributed problems and (2) Temporal planning problems. Indeed, many complex problems feature both of these. An example for each is briefly discussed: battle management for the spatial decomposition and mission planning for the temporal decomposition. A more detailed description of the hierarchical decomposition of Air Traffic Control problems, which has both spatial and temporal features, is described later in Section 2.2.

2.1.3.1 Decomposition for Spatially Distributed Problem-Solving:

A hierarchical decomposition can be viewed as a recursive implementation of the functional decomposition described in Sections 2.1.1 and 2.1.2 wherein the "system-to-be-controlled" is a lower level or a set of lower level work processes that are "controlled" or coordinated by an upper Master level. This hierarchical relationship is illustrated in Figure 2.5a. Note that each of the lower level work processes has identical structure to that of the upper level and, as we will see later, may have its "system-to-be-controlled" further decomposed into a set of work processes at a yet lower level. The most natural form of decomposition is one wherein there is a physical or geographic distribution of the entities at the lower level. The nature and implementation of the decomposition is strongly influenced by the availability of communications among the entities at the same level and between the entities at the lower level and the Master or superior level.

Example: Battle Management

The hierarchical decomposition of a complex problem is illustrated in the context of the notional near-land warfare battle management hierarchy shown in Figure 2.5b. This decomposition is notional in the sense that it is not intended to represent a solution to any actual problem. Indeed, all assets required for a complete description of the battlefield including surveillance and logistics are not contained in the figure. The figure does show four levels of a hierarchy, with the campaign operations management at the top and with the details of the air operations flowing down to individual aircraft at the bottom. Indeed, the G&C work processes for each aircraft at the bottom is functionally equivalent to that depicted in Figure 2.4. As suggested in the figure, the functional analysis described earlier in this section is applicable at all levels of this hierarchy.

This is a classical hierarchical decomposition that is based on both physical separation and functional aggregation. The physical separation of the entities is clear (air, land and sea). As is typical, the functional decomposition does not lead to completely independent entities, so that higher level coordination is required to harmonize the overall system solution.

2.1.3.2 Temporal Decompositions:

In contrast to the physical and functional disaggregation described above, temporal hierarchical decompositions are employed to simplify real-time, closed-loop planning problems [2.9, 2.10]. For these cases, the decomposition is characterized by higher levels that create plans with the greatest temporal scope (longest planning horizon) but with

the least detail. At lower levels, the planning horizon becomes shorter (nearer term), but the level of detail of planned activities increases. The less detailed plans at the higher levels coordinate or guide the generation of solutions generated at the lower levels.

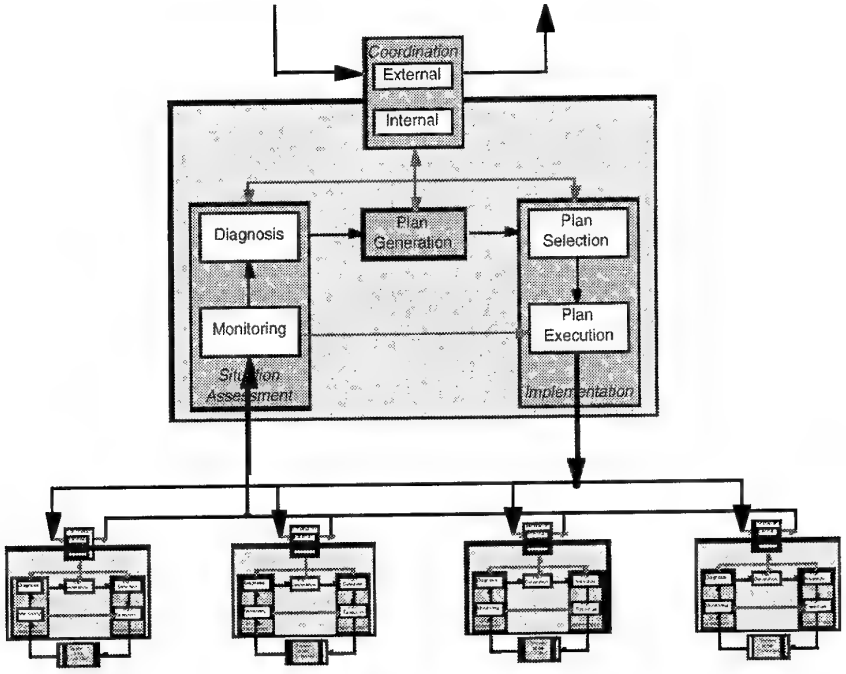


Figure 2.5a Hierarchical View: Distributed Problem-Solving at the Lower Level

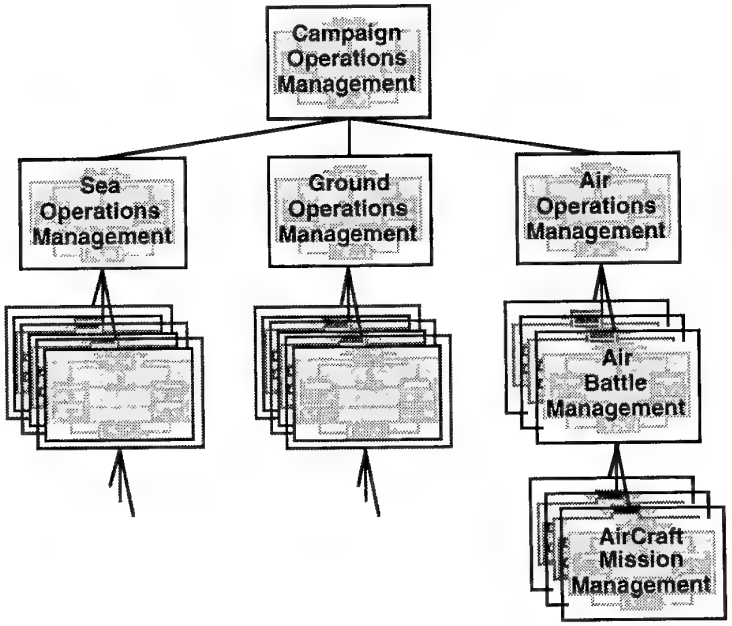


Figure 2.5b Near-Land Warfare Battle Management Hierarchy

Indeed, planning actions over extended periods of time at a high level of detail is typically both futile and impractical: *futile* because detailed actions planned on the basis of a specific prediction of the future may become obsolete well before they are to be executed due to an inability to accurately predict the future, and *impractical* because the computational resources required to develop detailed plans over extended periods of time may be prohibitive either in cost or availability or both. The relationship between the levels of the hierarchy and the planning horizon and level of plan detail is shown in Figure 2.6.

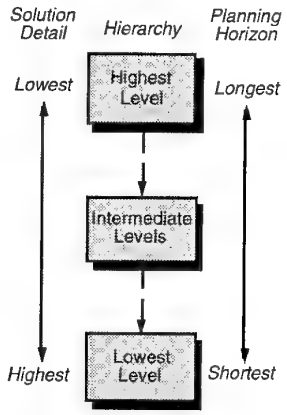


Figure 2.6 Characteristics of Solutions at Various Levels of the Hierarchy

Example: Mission Planning

For the onboard planning problem of a highly automated (potentially autonomous) air vehicle, mission and trajectory plans are developed within the hierarchy to optimize an established objective function (e.g., minimize fuel, minimize time or maximize a mission-specific measure of accomplishment) subject to specified constraints (e.g., allocations on mission timelines, fuel, flight safety, etc.). A typical hierarchical decomposition of the mission planning problem is one wherein skeletal plans of the entire mission are constructed at the highest level, the *Mission level*. The skeletal mission level plan must be generated at a sufficient level of detail to insure that onboard resources are sufficient to achieve the planned objectives and that timeline and survivability constraints are honored. At intermediate levels, the *Route/Activity levels*, near-term actions that are consistent with the mission level plan are planned in greater detail. Finally, at the lowest level of the hierarchy, the *Flight Safety level*, very near term commands are generated for sensor and control systems in a manner that ensures flight safety.

Figure 2.7 shows a two level decomposition of such a planning problem. The upper level creates mission plans spanning the entire mission and the lower level fills in the details of trajectory and payload activities that are required in the near term in pursuit of the mission plan. In the figure, the farther term plan generating entities at the second level are shown in gray to emphasize that although in theory the lower level would generate the detailed trajectory/activity plans for the entire mission, in practice only the near term plans are, in fact, produced.

2.1.4 Problem Solving

Thus far, the discussions of the functional decomposition of a work process have centered around applications of planning and decision-making. In addition, one can map a general closed-loop problem-solving scenario onto that decomposition. In this case, *plans* are expressed as *solutions* and the *system-to-be-controlled* is replaced by a *work object* (e.g., a machine tool fabricating a part, a robotic manipulator assembling a large space structure, etc.). Figure 2.1.1.8 depicts the functional analysis in this context of problem solving. The functions of each of the individual elements of the decomposition parallel those described earlier.

2.1.5 Cooperative Planning Agents

In many situations it may be either required or desirable for a team to create and execute the solution to a real-time problem. This implies that function allocation is constrained to reflect a team solution. Although the discussion presented here applies equally well to a person-person team, of particular interest is the person-machine team wherein a person and a computer act cooperatively in solving a problem. Supervisory control of an unmanned

vehicle falls within this class of problems. Figure 2.9 illustrates a modification to the functional analysis presented earlier that accommodates this cooperative approach. In that figure, we have assumed that only the Situation Assessment and Plan Generation functions are executed cooperatively. The entity on the left in that figure, makes an independent assessment of the situation and, if necessary, generates independently a plan or set of plans. Both the assessment and generated plans are shared with the dominant agent (shown on the right) who is responsible for selecting and implementing a single plan. Note that for the case of person-machine cooperation, either the person or the machine could be the dominant agent, and, as discussed earlier, the dominant role may reverse depending on the situation.

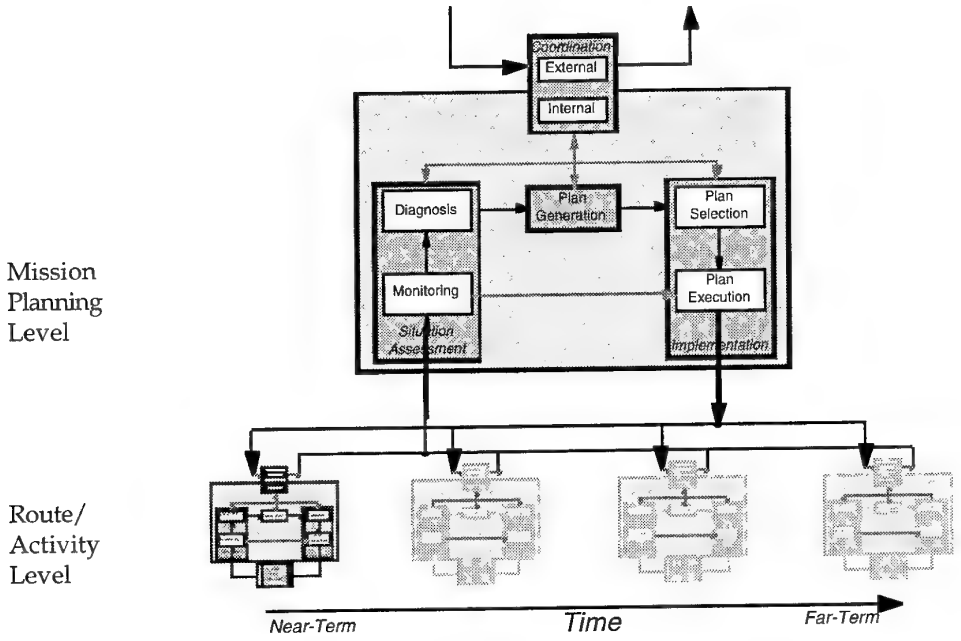


Figure 2.7 Hierarchical View:
Upper Level: Mission Planner, **Lower Level**: Route/Activity Planners

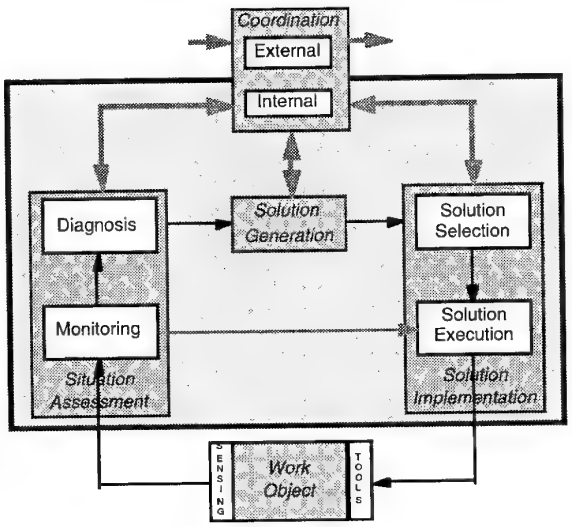


Figure 2.8 Functional Decomposition: General Problem Solving

2.1.6 Reference Following Control

The final illustration of the applicability of the generic functional analysis is for a classical reference following control problem. This class of problems represents an early example of the application of automation to a real-time, closed-loop work process. Figure 2.10 shows the mapping of each of the individual decision-making functions identified earlier onto the control problem: i.e., Monitoring, Diagnosis, Plan Generation, Plan Selection and Plan Execution. This mapping reinforces the view that the problems addressed in this report are a natural abstraction of those traditionally addressed in aircraft guidance, navigation and control. The ability to realize these abstractions in a machine implementation has been enabled by advances in computational hardware and algorithmic and knowledge-based systems approaches to developing solutions to the associated real-time problems that have been identified here. Section 4.5 discusses recent and ongoing advances in the development of non-traditional approaches to control law design that are based on fuzzy logic and neural (or connectionist) systems. These non-traditional methodologies are intended to provide an alternative to the traditional methodologies in addressing uncertainties in modeling the plant and the environment in which the plant is designed to operate.

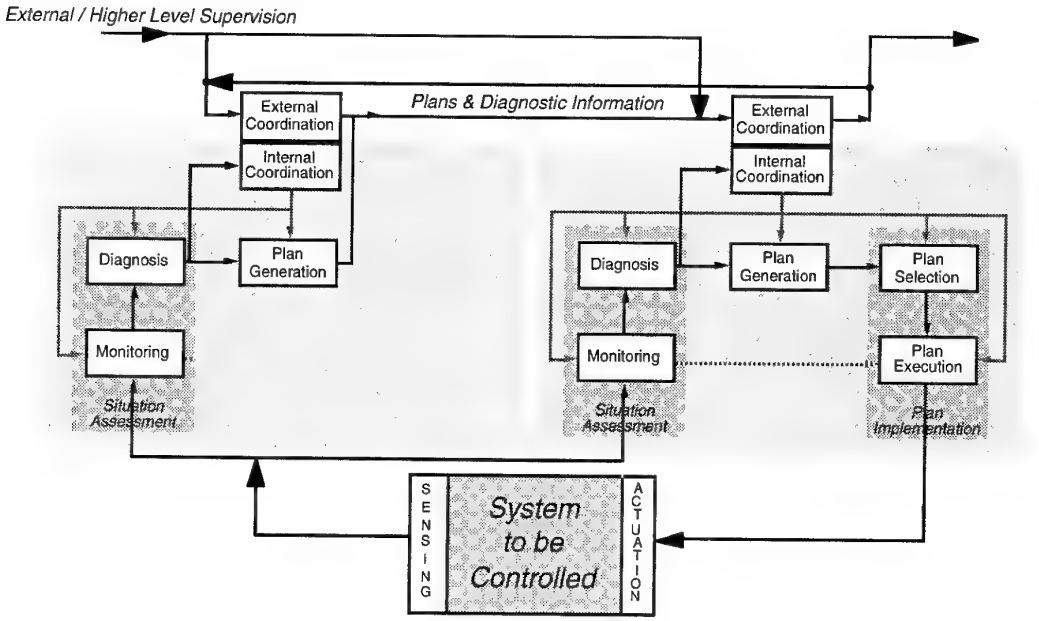


Figure 2.9 Cooperative Planning Agents

2.2 Refined Functional Structure for Applications

2.2.1 Aircraft Level and Navigation Level: Inner and Outer Loops

2.2.1.1 Introduction

This Section outlines a baseline structure for knowledge-based guidance and control functions at the aircraft level, i.e., for all functions performed within the inner and the outer control loops for a single air vehicle. This general structure is based on the type of structural elements of work systems/processes described in Chapter 1 and elaborated on in Section 2.1.

2.2.1.2 Hierarchical Structure Of Guidance And Control Loops

The overall guidance and control function can be represented in a hierarchy of functional levels with a number of cascaded loops. The outermost guidance and control loop, corresponding to the scenario control level, determines and commands the types of missions which serve overall objectives under given constraints. The commands for a certain mission and its pertinent objectives and constraints are passed to the next inner management and control loop, mission planning, for the derivation of a global plan for this particular mission. At the next level, the flight management of the selected mission is executed. Note that a number of different missions might be commanded and executed in parallel. For each mission one or a limited number of air vehicles are employed such that mission

management becomes identical to air vehicle mission management and control. Thus, there are (1) the scenario level, (2) possibly, an intermediate mission level usually on the ground (e.g., a mission command center or ATC), and (3) the air vehicle level. The latter is shown in Figure 2.11b

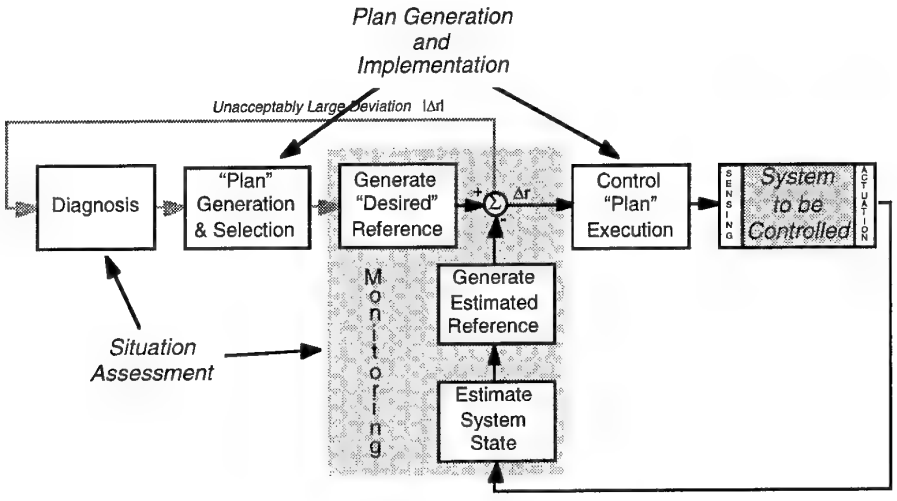


Figure 2.10 Functional Decomposition: A Control System Perspective

The air vehicle level outer loops can be further subdivided into onboard control levels for flight plan management, trajectory determination and flight path control. These are the loops where navigation plays an important role. The inner air vehicle control loop provides the vehicle attitude changes needed for the flight path control, by activating the control surfaces or other controls available.

The functions of the air vehicle level (or air vehicle work process) as a manned aircraft will be discussed in the following in terms of the underlying general functional structures. Mainly, the procedural representation of work processes discussed in Chapter 1 will be used.

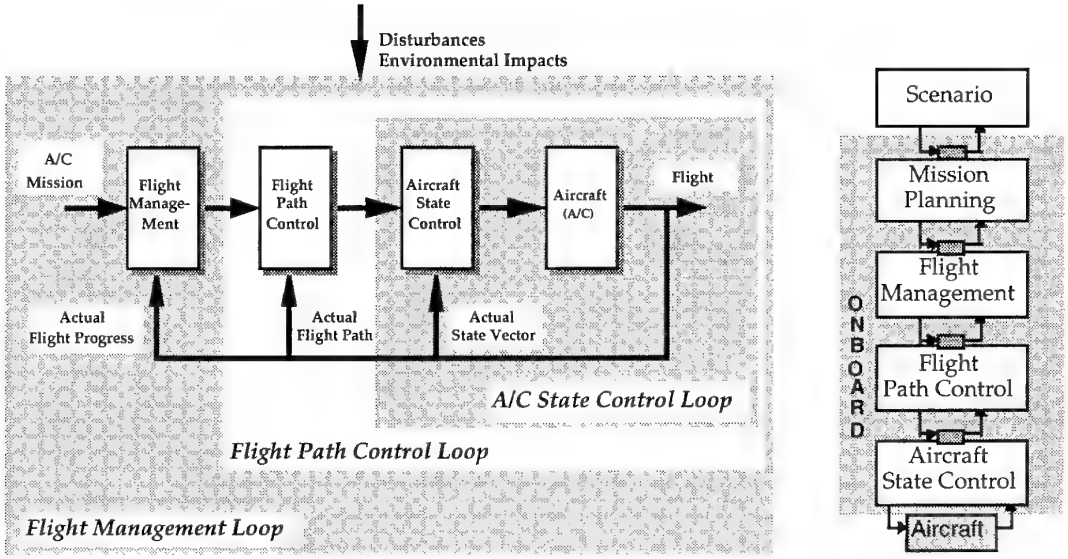
2.2.1.3 Baseline Work Processes For The Aircraft And Navigation Level [2.11]

Thus far, nothing has been said about the means by which the functions (work processes) corresponding to the individual levels or loops of the aircraft level, as shown in figure 2.11a, are achieved. In early aircraft, all of these functions were entirely executed by the pilot crew using few supporting tools, e.g., communications or sensor systems for acquiring information regarding internal vehicle or external mission situational parameters. Information processing was done by the pilot crew exclusively.

Today, the availability of tools to aid the pilot crew has been tremendously increased. The performance level and competency of the tools has steadily increased, particularly with regard to information acquisition and complex information processing. Consequently, the role of the pilot crew as operator has changed over the years, with a considerable share of onboard operations rendered to tools, i.e., the machine.

Thus, a variety of knowledge-based functions are becoming a domain of the machine. These machine-based tools are capable of assuming a large number of the functions in the hierarchical structure shown in Figure 2.11b. For instance, information about the overall dynamic situation of the mission, the aircraft, the environment and the pilot crew can be made available and assessed by such a tool. Of course, since the pilot crew has its own situation representation, not necessarily drawn from a work tool, it is essential that situational knowledge be shared between the pilot crew and the tools. This problem has the potential for being dealt with in the future by a knowledge-based cockpit assistant work process working cooperatively with the pilot crew.

A baseline structure of the knowledge-based work processes at the aircraft level is shown in Figure 2.12. The principal subfunctions of coordination, situation assessment, plan generation and plan execution are made explicit in the figure as well as the representation of the cockpit assistant function as a separate work process cooperating with the pilot crew and other interfacing processes.



(a) Traditional Control System View (b) Hierarchical System View

Figure 2.11 Aircraft- Related Functional Levels in Guidance and Control

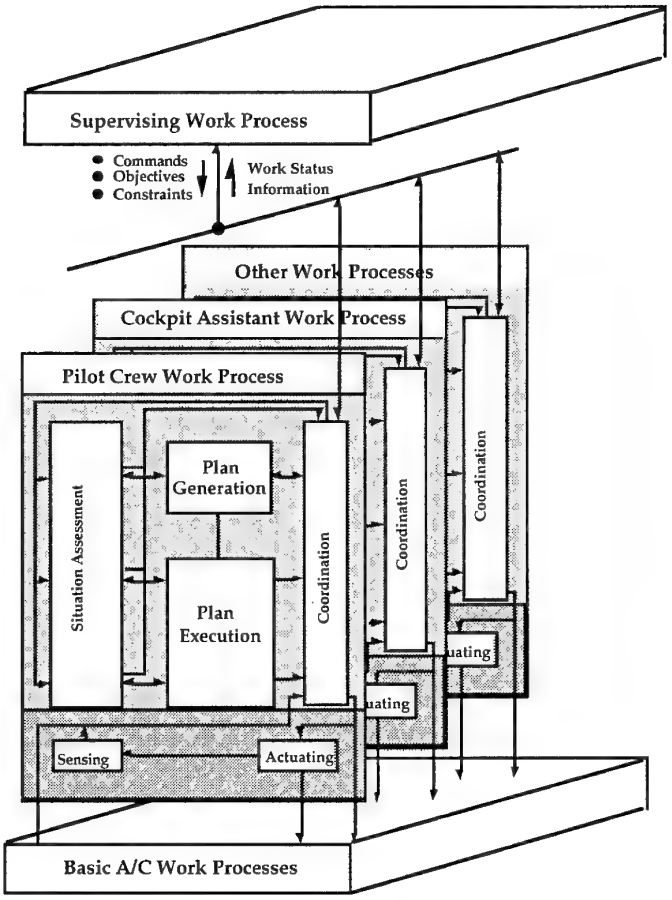


Figure 2.12 Global Structure of the Incorporation of Knowledge-Based Work Processes in the Aircraft Level

Figure 2.12 shows, in addition to the pilot crew work process, the other work processes within the aircraft level as well as those outside the aircraft at the supervising functional level (e.g., air traffic control (ATC)). The cockpit assistant as a knowledge-based onboard work process interfaces with the supervising function, the pilot crew, the basic aircraft work processes (in particular those for sensing and actuating) and possibly with other onboard work processes. Among those, the pilot crew work process is the most prominent. The interface between the pilot crew and the cockpit assistant must be chosen carefully in order to ensure effective handling and command delivery as well as effective modes for communicating information that is provided by the cockpit assistant functions. In this regard, speech input and output are crucial .

Information regarding environmental processes detected by sensors onboard the aircraft or on the ground or through communication which might affect the aircraft must be accommodated by the above mentioned interfaces. These external environmental processes include: weather, other air vehicles and military threats.

Cockpit Assistant Work Process

Looking further at the subfunctions of the cockpit assistant work process, the interface functions, including those responsible for suitably formatting information for the pilot crew and for controlling the flow of information are performed by a knowledge-based *coordination* function which is an element of each of the work processes (see also Figure 2.4). The coordination function controls the internal distribution of the information received from external work processes and thereby coordinates the other cockpit assistant functions, subject to supervisory objectives and constraints. The coordination function also determines the content of the information flowing out to the surrounding work processes, including the presentation of advice and messages to the pilot crew.

The information which is used by the coordination function in the derivation and forwarding of messages to the pilot crew is drawn from internal functions including *situation assessment, plan generation and plan execution.* Coordination is at the heart of the key purpose of the cockpit assistant process, normally generating advice and messages for the pilot crew, and the other functions can be viewed as service functions for this purpose. These autonomously track all occurrences relevant to the evaluation of mission accomplishment through situation assessment (monitoring and diagnosis or situation interpretation), they prepare alternative plans, problem solutions and decisions on the basis of the situation assessment and they provide a reference for plan execution performance evaluation. As such, they represent a machine capability for carrying out the functions that the pilot crew is trained to perform.

The design criterion for these service functions might be either to mimic the pilot crew performance as closely as possible or to search for the globally optimal performance, possibly beyond the pilot crew capabilities (see also [2.12]). Careful study is required to determine when the latter criterion is both feasible and beneficial. For instance, for certain cockpit assistant functions such as the generation of plan recommendations, it can be desirable to search for the absolutely best solution, subject to specified objectives, constraints and given knowledge of uncertainty in the assessed situation. A search for the optimum is not likely to be performed equally well by the pilot crew due to mental restrictions such as working memory limitations (see Chapter 3). On the other hand, machine performance close to that of the pilot crew might be important, for instance, when monitoring the performance of the pilot crew or when predicting actions by other cooperative (human) agents.

From a different perspective, these service functions intrinsically represent the capability to fly the airplane autonomously. Indeed, there are applications, where this might be desirable as an option on request of the pilot crew. Thus, for the sake of both its advising and monitoring role and its autonomous flight capability, the cockpit assistant process should have the capacity to perform all tasks or functions which the pilot crew performs, in particular for

- Situation assessment
- Flight plan generation and decision support and, possibly, decision making
- Execution of plans and decisions.

In summary, the structure of the cockpit assistant work process in Figure 2.12 supports all guidance and control loops as shown in Figure 2.11. It also emphasizes the importance of the coordination functions for the various agents such as the pilot crew and cockpit assistant, depicted as separate work processes in Figure 2.12, in ensuring synergism in the accomplishment of situation assessment, plan generation and plan execution for all guidance and

control loops taken as a whole. Many of these processes or subprocesses are appropriately implemented as knowledge-based in the sense as defined by Rasmussen [2.13].

The functions not yet discussed in detail, are the plan generation and plan execution functions. The plan generation function supports long/term as well as short/term decisions which must be made in order to resolve the problems detected and identified by situation assessment. The plan generation function is a knowledge-based function. Plan or problem solution alternatives are derived on the basis of background knowledge about the mission goals and are evaluated based on established goal relevance criteria. Also an important role for this function is to ensure coordination between the planning performed by the pilot crew and the planning performed by the cockpit assistant.

Once a plan has been generated and selected, the plan execution function comprises either rule-based or skill-based services which are made available for carrying out the decision, i.e., the selected plan.

The main purpose of the knowledge-based cockpit assistant is to ensure that support to the pilot crew will be ready at all times and will be offered when needed (see Chapter 3). It must know and account for the pilot crew needs and intentions. To this end, it must be able to understand and evaluate the actions of the pilot crew. Consequently, most of the cockpit assistant functions are based on a continuous stream of information about the behavior and expected performance of the pilot crew. The expectations are based on a normative pilot crew, and the behavioral instantiations are for those of the actual flying pilot crew. Also, human factors and condition parameters that are used to model the actual values of pilot crew resources should be incorporated as well as models of pilot crew intentions and error contingencies. Since the pilot crew actually flying the aircraft is neither required to nor able to provide this information, the cockpit assistant process has to contain it or generate it on its own.

This very crucial knowledge component becomes available by means of a dynamic model of the pilot crew (Figure 2.13), covering all of the above-mentioned aspects (see Chapter 3). The application of the model can be carried out separately for each of the cockpit assistant functions. An alternative approach is to have each function draw the information necessary for its particular needs from a single pilot crew model / representation accessible to all cockpit assistant functions. The kinds of functions to be considered in that model include those discussed earlier, i.e.:

- Situation assessment
- Plan generation
- Plan execution

Of course, these functions as performed by the pilot crew will not necessarily have outcomes equivalent to those of the corresponding cockpit assistant modules. Thus, the behavior models will primarily yield the pilot crew actions, possibly in the form of ranges for the actions. For the determination of these ranges, objective safety limits are taken into account.

Since the information about the performance of both a normative crew member and that of the actual flying crew member must be accommodated by the model, several dynamic pilot crew models are to be represented in parallel:

- Normative behavioral characteristics (with and without mismatching contingencies)
- Individual characteristics, condition and behavior (with and without mismatching contingencies)

By employing inference mechanisms along with the outcomes of these models, prediction of crew member actions as well as recognition of their intents can be achieved. This, in turn, can be used to detect discrepancies with respect to accomplishment of the mission goal and subgoals and to detect pilot errors, possibly caused by mismatches due to certain cognitive deficiencies or because of violations of limits of the principal human capabilities. In this context, the model of the individual crew member condition and behavior is used in the decision process and for scaling the warning messages to the crew in case of anticipated discrepancies between agreed mission goals and crew actions. *Obviously, the better the crew model, the more relevant and effective the assistance provided by the cockpit assistant process.*

2.2.1.4 Situation Assessment In The Aircraft Level

From the discussion about the structural elements of the aircraft guidance and control loops shown in Figure 2.11 and their knowledge-based processes shown in Figure 2.12 it can be argued that situation assessment is the basic enabling functional element regardless of whether it is performed by the pilot crew or by a work tool like the cockpit assistant work process or by both collaboratively.

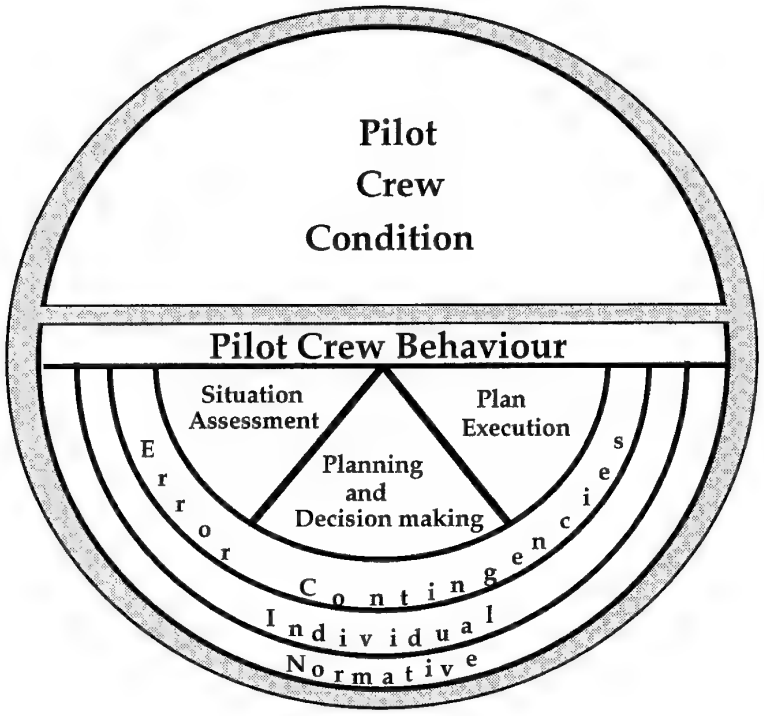


Figure 2.13 Elements of a Dynamic Pilot Crew Model

The access to situational information and its representation plays a highly important role. The output of a given loop becomes a source for situational information in the sense that it becomes the command input for the next inner loop. Additional inputs for situational information are both direct feedback and outputs of loops further in. This feedback includes not only raw sensor data but also higher level information provided by interpretation of sensor data in the context of the mission situation and the progress toward the goals of the level. Without this information, nothing would work properly. If the ability to achieve situation awareness were lost, the aircraft would immediately be in danger.

Therefore, in focusing on situation assessment in the cockpit assistance work process , the structure of this function will be discussed in more detail in the following, with a further examination of its central and comprehensive character.

The situational status can be represented by feature elements pertinent to the situation at each point of time including elements reflecting situation changes from times before and after a given point in time. The actual instantiations of the situational status can be thought of as being kept available for any work process in dynamic databases or in a single central database (which also contains all relevant static data), as is depicted in Figure 2.14. This database can be considered as passive, being fed with the pertinent data by the surrounding work processes. Conversely, this database is utilized by the work processes in drawing from it all situational data needed for their specific functions, i.e., for instance, for instantiations in their own knowledge bases. Consequently, the function of situation assessment consists of:

- Determination and instantiation of elements of the current situational status
- Interpretation (monitoring and diagnosis) of the situation in the context of the mission goals

The work processes of the cockpit assistant function shown grouped around the database kernel in Figure 2.14 can be broken down into those for: interpretation, plan generation, plan implementation and coordination.

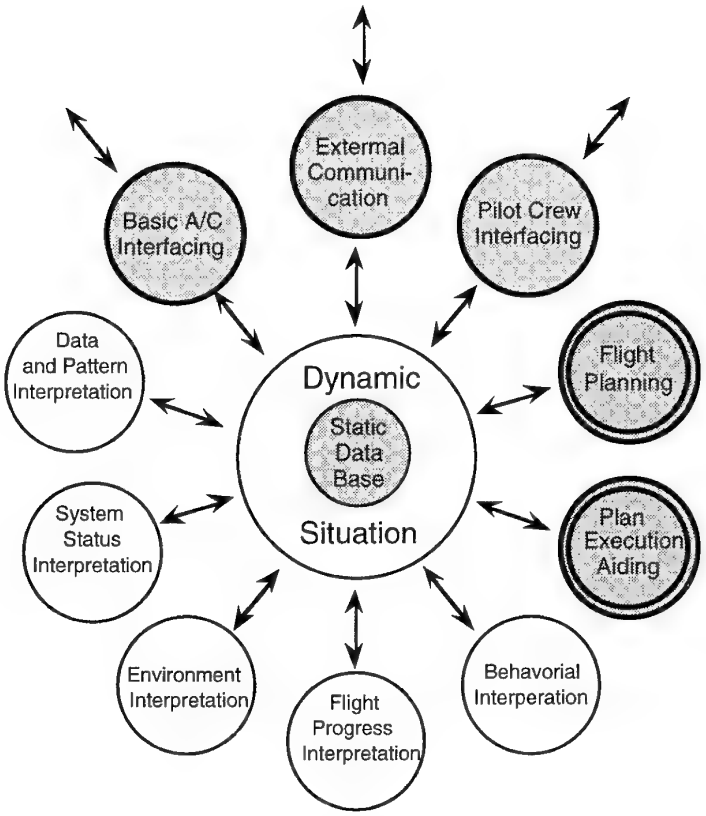


Figure 2.14 Guidance and Control Work Processes in the Aircraft Level and Their Contribution to Situation Assessment

Five interpretation processes are depicted in Figure 2.4:

- the process for data/pattern interpretation such as data fusion (e.g., fusion of navigation data from different sources) or interpretation with regard to machine perception,
- the monitoring and diagnosis of the aircraft system status
- the interpretation of environmental incidents like weather events or traffic events not previously announced by means of communication
- the monitoring and diagnosis of progress toward the flight plan
- the monitoring and diagnosis of the pilot crew condition and behavior, including the recognition of the pilot crew's intentions

The interpretation processes draw information from the data base about a great number of static and dynamic situational elements including aircraft state of motion such as position and attitude, aircraft system parameters such as displacement of controls or system status parameters, environmental parameters such as weather and traffic data and also parameters representing the pilot crew condition, actions and intentions. The interpretation processes condense this information subject to certain specifications, thereby generating higher level situational information such as indications about deviations of the mission plan or actual changes in risk or danger level. Depending on the degree of interpretation complexity, some of these belong to the category of knowledge-based functions, which draw on reasoning capability and sophisticated knowledge representations, including knowledge about the pilot crew as described in Section 2.2.1.3 and Chapter 3. The data drawn from the database are used for the necessary instantiations in the knowledge representations of the interpretation process. By feeding the interpretation results

back into the database as higher level situational data, i.e., enriching the up-to-date situational picture, these interpretations become available to all other processes.

In addition to the interpretation processes and the work processes that provide direct assistance to the pilot crew like flight plan generation for onboard mission management and plan execution aiding, three coordination processes are depicted in Figure 2.14

- communication interfacing with external agents,
- interfacing the cockpit assistant functions with the other aircraft system functions, in particular its sensory functions and
- interfacing with the pilot crew by providing intelligent, multimodal information flow management.

To some extent, all work processes, either as their prime purpose or as a subtask, participate in the function of situation assessment, i.e., keeping the database content up-to-date. The interpretation processes serve the situation assessment function exclusively. The remaining processes make use of these data for functions other than situation assessment. However, they also serve the function of situation assessment in conjunction with their main function through rendering their outputs for instantiations in the situation representation. This is true for the outputs of the coordination processes and the assistant processes for flight plan generation and plan execution aiding.

The situation assessment function, as described above, provides a good scheme for the development of the situation representation subject to the information needs of the guidance and control processes. It also becomes evident that the situation representation has distinct subgroupings and levels of aggregation and abstraction. At the same time, it shows that all functions of the cockpit assistant work process are involved to some extent in the situation assessment function.

2.2.1.5 Concluding Remarks

A baseline structure of the guidance and control work process in the aircraft level is outlined with a focus on knowledge processing. In future guidance and control of air vehicles, knowledge processing will be shared by both the pilot crew and work tools like cockpit systems and associated equipment the pilot is using.

A prominent work tool of this kind is the cockpit assistant. The underlying work process is characterized as a systematic approach to ensuring the functional symbiosis of knowledgeable work tools and the knowledgeable operating system: the pilot crew. The two are unlikely to have equivalent knowledge representations. By means of the cockpit assistant process there is the chance that they can cooperate in a way to optimally carry out the flight mission.

The keys for successful integration of automated high level guidance and control functions such as mission plan generation are properly performed situation assessment and ensuring that both pilot crew and assistant system know about their differences in situation assessment. In particular, if this can be ensured on the side of the cockpit assistant, awareness of an urgent need of the pilot crew can be achieved no matter whether the need is caused by events overtaxing the pilot crew or by some kind of pilot mismatch. In either case, the knowledge-based cockpit assistant can offer sensible support. For the time being, this cannot be accomplished in a comprehensive manner. However, it has been shown by development programs (see Chapter 6) that an effective increase in overall guidance and control performance can be achieved with intermediate steps of knowledge-based assistance.

2.2.2 A Functional Description of Military Air Operations

2.2.2.1 Introduction

In this section, a plan oriented process model is developed to describe the dynamic relationships between military air operations at all levels of command. This model is mapped onto the functional model developed in Section 2.1, and these functions are specified in general for the military air operations domain and, in particular for military aircraft missions. No attempt is made to identify or distinguish human or machine function in this study. Rather, the focus is on understanding the functional requirements of the complete work-system. The potential for applying knowledge-based system technology to these functions is discussed in general terms.

2.2.2.2 Process Model of Military Air Operations

Military air operations are essentially plan oriented. At all levels from commander to combatant and in all domains, planning is the fundamental organizing principle and is the key to solving problems relating to uncertainty

and confusion in combat. Thus, a planning paradigm is a "natural" representation of the full scope of military air operations. Such a paradigm is adopted here to provide a framework for a discussion of G&C functions in military air operations (perhaps more appropriately referred to as command and control functions) and to illustrate functional relationships between military air operations at levels ranging from theater to battlefield to individual missions.

The military notion of a "plan" evolves from a process that includes: (1) situation analysis; (2) determination of objectives; and, (3) selection of a course of action intended to realize the objectives. The process is viewed as iterative along a temporal axis and recursive at finer levels of detail across the hierarchical command. Thus, the generation of a plan at the highest level ("campaign") invokes the formulation of subordinate plans through a similar process at lower levels, in order to meet goals implicit in the planned courses of campaign level action. For example, in Figure 2.15, at the campaign level, a course of action is selected that calls for the defeat of enemy fielded forces. This course of action implies an objective to gain air superiority over enemy territory. That objective then generates a course of action indicating a sequence of combat actions against enemy air defenses. The objectives of these actions then define aircraft missions in the daily battle plan. Mission objectives finally result in the selection of targets for individual aircraft and mission plans are then crafted to create a desired effect on the targets. Thus, consistency of objectives across levels and coordination at every level in the command structure is effected, largely through the inheritance of plan objectives.

The intermediate levels in the command hierarchy exist primarily to provide insight at the appropriate scope into important operational detail. Senior commanders formulate the general concept of operations, but the mass of significant detail required and the lack of information at the Senior Commander level have made it impractical - probably impossible - to build plans beyond the level of detail required to "command" the next level of planning. Historically, considerable initiative has been exercised at each level in the hierarchy in the formulation of operational plans. Interestingly, while the desirability of and degree of latitude given to initiative are central issues in the doctrinal debates that distinguish various military organizations throughout the world, actual practice - especially in air forces - seems to vary only slightly. No doubt, this similarity has heretofore resulted as much from practical limitations in the collection and interpretation of operational data as from organizational theory. No matter how strongly senior officers have felt about the dilution of their intent as it passes through the command structure, they have simply not been able to know enough about the battlefield to make useful decisions in a timely manner. Significant advances in digital battlefield communications infrastructure and computer aided situation assessment are likely to change this in the future.

Plans evolve (at each level of the military hierarchy) in an iterative fashion in response to events that: (a) alter the situation; (b) add or delete objectives; or (c) indicate a different a course of action. Figure 2.16 illustrates the relationship of plans at various levels in the military hierarchy in this process.

In military terminology the distinction between "planning" and "execution" is somewhat ambiguous. In general, the implied notion is that planning is deliberative and execution is active. In fact, the distinction is arbitrary. Indeed, a plan may be executed by either the creation of a subordinate plan (series of plans), or the creation and selection (instantiation) of a course of action. Thus, at the Theater level, a campaign plan is "executed" when a Master Attack Plan is created. The Master Attack Plan is, in turn, executed by the creation of an Air Tasking Order. The Air Tasking Order is executed by the creation of a Mission Plan which is then executed by the creation of engagement plans (tactics) and ultimately, a launch signal to a weapon.

From a dynamic perspective, a plan is considered to have been executed when it is complete. Thus, a mission plan is executed when the aircraft has safely returned to base (and thus "is executing" when the mission is being flown). The important aspect of this is that military plans should be viewed as constantly evolving. Thus, a plan is thought of more as a preferred sequence of actions leading to success in achieving the ultimate goal. Military plans are not typically seen as the optimum or only way to achieve the goal or even as a detailed description of what is likely to transpire. On the contrary, plans are notoriously volatile in the conventional military wisdom. A frequently heard phrase - "the plan evaporates on first contact with the enemy" expresses the common view that plans are established as flexible guidelines that organize activities toward desired objectives and are not seen as rigid prescriptions or mandatory procedures. Considerable emphasis is placed on plan "flexibility" or the necessity to adapt, modify or alter a plan or switch to a new plan when circumstances indicate. Thus, military planning is part of an ongoing "real-time closed-loop" process comprising the 5 generic functions outlined in Section 2.1, namely: monitoring, diagnosis, plan generation, plan selection and plan execution.

Military plans are central to survival and success in the chaos of battle. Thus, there exists both a reluctance to abandon and, if distracted, a strong attraction to return to the general scheme of a plan in combat action. Thus, the

typical reaction to unexpected events is first, repair the plan, next modify it if repair proves impossible, and finally, after all else, re-plan. This aspect of military planning is seen at all levels in the command structure and is so important that combat itself is sometimes characterized as a battle of wits whose objective is to force the opponent to conform to one's own plan while abandoning his plan. This is, in fact, the principle illustrated in the primacy of the offensive and the general dislike for reaction in military theory.

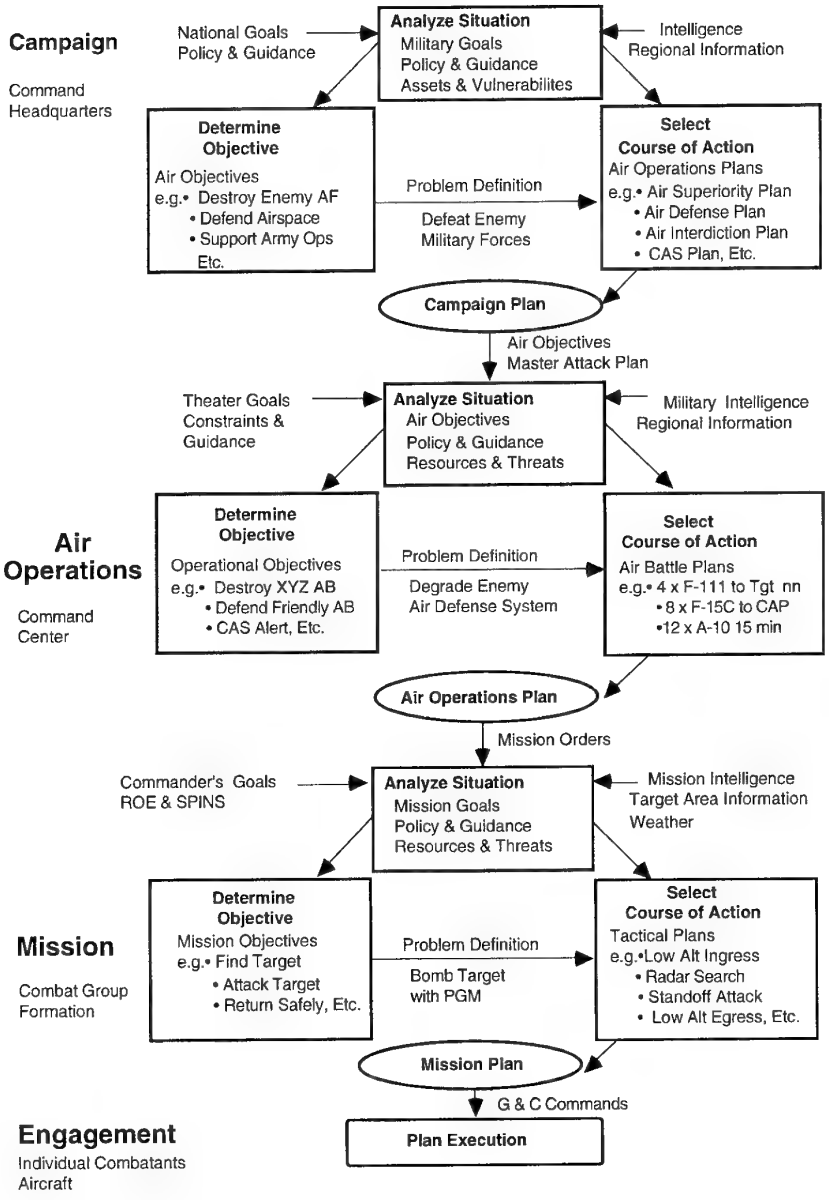


Figure 2.15 An Air Operations Process Model

In recent years, considerable attention has been focused on the joint problems of improving and speeding the planning cycle. Several factors influence this trend. First, improved data collection and information dissemination systems, and especially space-based sensors, have offered senior commanders more direct insight into battle area

issues. Secondly, modern communication systems have greatly improved the ability of these commanders to directly influence a battle. Third , dramatic improvements in weapons systems allow very precise and efficient application of force and indicate explosive growth in the kinds of effects that can be achieved. The combined impact of these trends and consequent importance of good planning was clearly visible in the 1992 war with Iraq.

Thus, the notion that a well-planned, accurate application of air power can quickly lead to victory - and, by implication, to rapid defeat, if the application is by the enemy - has motivated numerous efforts to **automate and integrate** planning and execution functions at all levels of military air operations.

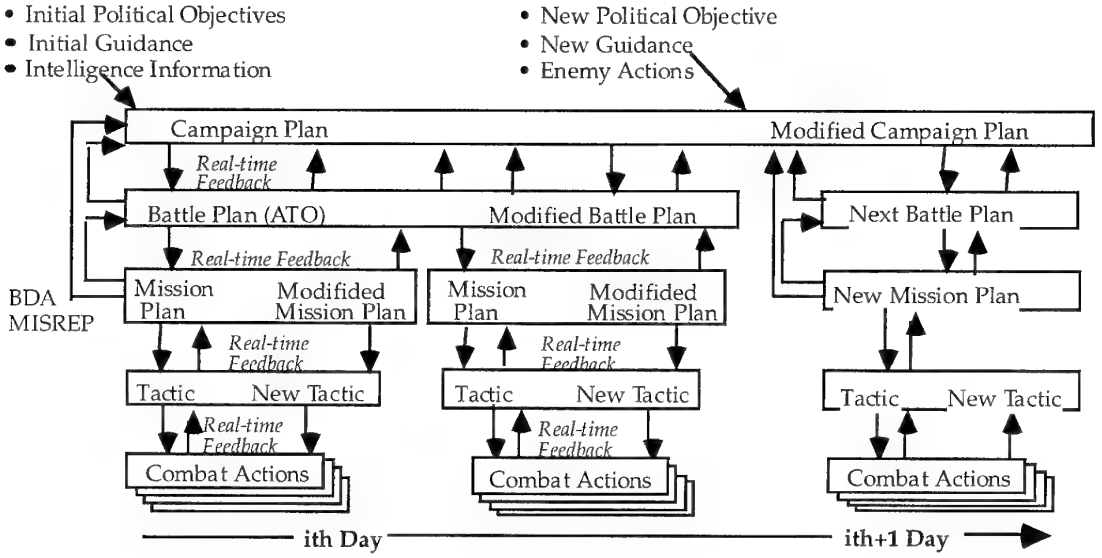


Figure 2.16 Evolution of Military Air Operations Over Time

2.2.2.3 Functional Model of Military Air Operations

From this perspective, military air operations may be conveniently mapped onto the functional model developed in Section 2.1. (See Figure 2.17) Note that in this view the recursive nature of the process is conserved and the "System to be Controlled" for a given level of interest becomes the entire structure below that level.

Thus, if we examine the model at the Senior Officer's level, the "System" is the Command and Control System for the Theater. At a Command Center level, the system is the collection of assets that constitutes the force committed to battle. For combatants, the "System to be Controlled" then, is the aircraft or weapons system at their command.

With reference to the diagram in Figure 2.17, the general a functional cycle may be said to begin at "Coordination with External Agents." At this point, tasking is received, directing some activity. The tasking may be precise, or, more typically, general. If it is precise - that is, directing or requesting a particular action - it is traditionally expected that the tasking will trigger the specification of a previously prepared partial plan or set of partial plans. This sort of tasking may take the form of the initial phase of an offensive, or an air defense "scramble" order, or at the combatant level, an order to execute a particular tactic. In such cases, temporal constraints - imposed by either the need for rapid reaction or for precise coordination of multiple actors - imply the prior existence of a prepared course of action and coordination is limited to acknowledgment of the order and subsequent status reports.

In such a case, some function, here referred to as "Internal Coordination," directs the tasking to a function responsible for the determination of which plans are to be executed - "Plan Selection" in our scheme. An appropriate class of prepared plans are then specified or instantiated to accommodate the actual conditions and the

best among them is selected for execution. "Plan Execution" then performs functions related to organizing and sequencing selected plans and directing the implied tasking to appropriate elements of the "System to be Controlled."

More typically, however, general tasking is received. Such tasking is transmitted in the form of objectives, constraints (policy, rules of engagement, etc.) and accompanying situational information. In this case, both Internal and External Coordination may be more complex. In our model "Internal Coordination" directs the tasking to a functional partition responsible for refining the problem definition, here named "Diagnosis." The objectives contained in the tasking are then interpreted and compared to existing system goals. If no match is apparent, "External Coordination" must communicate the difficulty to the tasking agency and request clarification or refinement of the tasking.

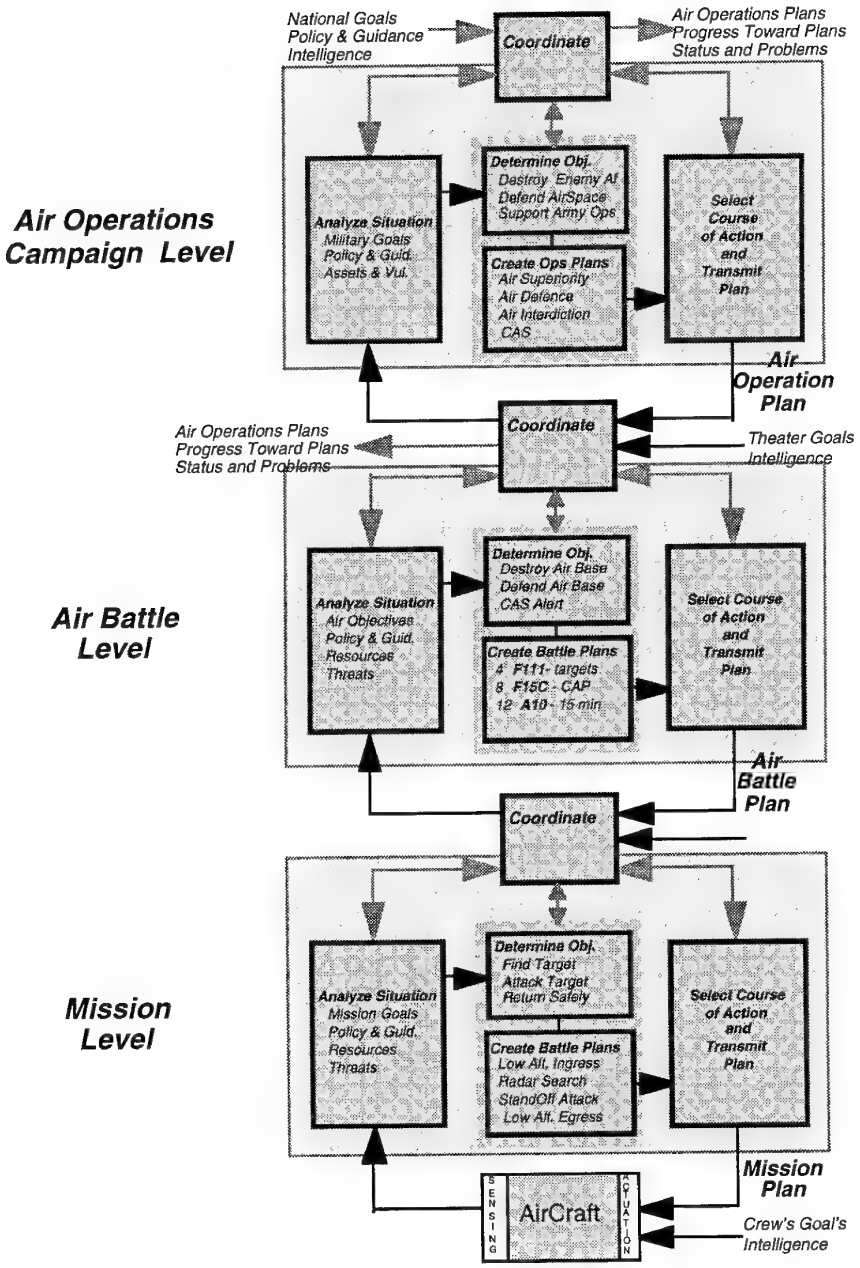


Figure 2.17 Functional Model of Military Air Operations

When a match to system goals can be made, an initial diagnosis is accomplished to define the problem or problems that must be solved in order to accomplish the task. In general, such a diagnosis consists of a comparison of the actual or projected state of the world to the desired or goal state and a characterization of the difference in terms of potential system actions. Such an analysis typically begins at a coarse level to determine general capability and proceeds to a review of classified plans to determine if a suitable plan currently exists or can be constructed.

If a negative result is obtained at this point, support from cooperating agents may be requested or, failing this, new tasking may be negotiated with the tasking authority. Of course, if no relief is provided at either level, the problem diagnosis is then repeated with an adjusted set of evaluation parameters.

Once the problem has been defined, "Internal Coordination" may then direct it to "Plan Selection" if potential solutions are believed or known to exist or to "Plan Generation" if not. Existing plans that may satisfy the tasking are then specialized and tested to determine feasibility with respect to system resources and expected or actual capability. Feasible plans are evaluated with respect to established measures of effectiveness and the best of these is selected for execution.

If no plan that meets the demands of the problem is found, a new one is created in "Plan Generation." This function decomposes the objective posed by the task into a list of subordinate goals, consistent with defined constraints, that, when completed, will result in the achievement of the objective. Further problem analysis is conducted to illuminate risks and opportunities and suggest solution strategies. Guided by these strategies, goals are then matched to requisite actions and sequenced or otherwise ordered to provide a suitable plan for the solution of the defined problem. The result is then tested for feasibility and evaluated to assure that at least a minimum level of satisfaction will be reached. Obviously, if such a plan is not generated, further "External Coordination" is called for.

Once a plan is identified as the preferred solution to the problem, it is passed to "Plan Execution." At this point, potentially competing plans are integrated into a coherent sequence. Subordinate plan actions are then directed as tasks to elements of the "System to be Controlled."

The results and effects of these actions by the "System to be Controlled" on both the external world state and the internal system state are observed and interpreted in the "Monitor" function. Some predictable effects - emergency or contingency situations, for example - may automatically trigger a prepared plan. Others are passed to the "Diagnosis" function for assessment. At this point in the cycle, the actual state is compared to the expected state and the problem space is redefined. Typically, trend information and current intelligence information is used to both improve projections of the current state and to evaluate the essential hypotheses of the plan. If it appears that the basic hypothesis is flawed, a new plan is developed or selected. Deviations at a lower, sub-goal, level result in modifications to the basic plan, while lesser differences necessitate only a re-specification of the original plan in the subsequent cycle. Thus the system can be seen to progress toward the eventual realization of the assigned objective.

2.2.2.6 Functional Description of a Special Case: Pre--Mission Planning for Aircraft Missions

A special case of the functional model for military air operations is seen in its application to the analysis of Pre-Mission Planning activities. It is distinct from the general case in its that pre-mission planning is "open loop" and the functional cycle actually completes with the formulation of a plan. The plan is executed with no direct feedback to the process, regardless of whether the plan succeeds or fails. Thus the emphasis is on a deliberate planning process and on reasoning about the uncertainties in the data and the planning assumptions. Humans who bring judgment, experience, creativity and intuition to the process, frequently experience difficulty in objective evaluation of uncertain data. Knowledge-based systems offer the potential to add formality and objectivity to the process and to allow the consideration of greater and more current data in order to reduce the risks inherent in uncertainty.

Military aircraft missions are invariably guided by a mission plan. A mission plan is a sequence of actions, motivated by a particular list of sub-goals, that can satisfy a mission objective within the bounds of applied constraints. Strictly speaking, such plans are "partial plans" that are intended to be instantiated when a defined set of conditions exist. These plans then form a list of preferred actions and responses for the work-system as it executes the mission. In current practice, these plans tend to be limited by (1) aircrew experience; (2) a general consensus among aircrews that potential effectiveness must be sacrificed for simplicity; and (3) pre-mission planning time. Improvements in the work system might therefore be found in an ability to develop richer partial plans that are better suited to particular mission conditions, and perhaps, that offer more and finer-grained options.

A Mission is performed by an aircraft work-system to accomplish a set of objectives. These mission objectives are a complex set of goals which have either been assigned by a higher authority or are inherent in survival or the

designed role of the aircraft. Mission objectives may also be described as state abstractions that prescribe significant aspects of some desired world state expected to exist at the conclusion of the mission. In fact, these objectives are interpretations of objectives inherited from or mandated by another agent. One important effect of an improved work system in this domain might be found in minimizing discrepancies between the promulgated objectives and the internal, interpreted objectives.

Mission plans are prepared, formally or informally, by aircrews before every mission. For many modern aircraft, mission plans are developed with the aid of Mission Planning Systems [2.14, 2.15]. Various Ground-based versions of such systems are under development in an attempt to seek improvements in the work-system as described in the previous paragraph. Currently, considerable thought has been given to providing some version of this capability to aircrews in the cockpit. On the other hand, the most common form of mission planning today is a manual process, carried out by aircrews on the ground. This process typically consumes periods ranging upwards from two hours and is viewed as a "bottleneck" that tends to restrict the desired pace of air operations in modern combat. In any case, pre-mission plans are the essential link between the commander's concept of operations and battlefield actions.

Planning and executing a mission for a military aircraft can be viewed in the context of the functional decomposition developed in Section 2.1 where the "system to be controlled" is the mission itself, not a physical system (the aircraft). In general, the "Monitoring" function acquires data about the scenario considered through the information system, while "Diagnosis" assesses the risk distribution throughout the scenario. Such data will be used by "Plan Generation" which plans missions acquired from the user, thus allowing "Plan Selection" to eventually simulate the mission, and to choose the most promising solution. The selected result is a fully elaborated, but unspecified, partial plan that will be used by pilots to execute the mission. The prepared mission plan thus becomes the essential component of the basic internal model for the realization of the mission. Mission execution will then affect the environment in a way that will be perceived by the information system and this new data may be used in a following cycle to plan a new mission.

As described above, the time needed for one control cycle equals the time needed for planning and executing a mission. On the other hand, the mission's control cycle could vary according to its frequency. Thus, if the possibility of several control cycles per mission is considered, we may then see a potential for stronger ground control of the mission in progress, i.e., a stronger link between the ground and the aircraft.

In this scheme, the function called "Monitoring" gets data from the environment where the mission will be executed. This data, seen as external state signals, includes environmental information, such as, weather and number and position of enemy threats.

This data input phase is carried out before starting the process, and involves many information sources - intelligence, policy, doctrine, and such. As seen in the general model, data is not interpreted here, but simply collected, classified and sent to the "Diagnosis" function. At that stage of the process, the environmental data is evaluated and risk levels for each zone of the scenario are computed.

The Coordination function directs internal and external coordination. The External Coordination function takes care of the interaction with other systems at the same or higher level of the hierarchy (i.e. battlefield management systems and other pre-mission planning systems at the same level). It collects high level objectives, constraints and other information. Interaction with other Pre-Mission Planning systems adds considerable complexity because all the solutions that may be adopted must be negotiated, perhaps with recourse to a higher authority for decisions.

The "Internal Coordination" function manages internal planning activity for the unit or formation, and, when required, coordination of plans for individual aircraft within the formation. This function receives inputs relevant to the final objective, constraints, and strategies required. The eventual output is a mission plan for each aircraft involved in the action.

The "Plan Generation" function is viewed as a composition of three sub-functions: Attack Planning, that plans the critical attack phases; Path Planning that plans transfer phases; and Mission Management that coordinates the work of the two other functions.

Attack Planning (AP) is the most crucial activity, and it is performed during the first phase of the whole process. This function establishes the best direction of attack, the best attack tactic and, finally, the best egress from the target zone. Other mission phases are then planned accordingly.

Path Panning (PP) is essentially route planning, and is generally considered a transfer phase. This kind of planning is dominated by "threat avoidance" considerations, and tactical factors such as the pilot's desire to hide his intentions from the enemy and avoidance of predictable routes. The Mission Management function then compares and combines the partial solutions provided by AP and PP for the final mission plan.

After the generation of a set of possible plans for the mission, the Plan Selection function makes a final choice by means of criteria that are suggested by the "Internal Coordination" function. At this point, the prepared mission plan is transferred to the aircraft system for execution.

2.2.2.5 Further Expansion of the Functional Model at Aircraft Mission Level

The following section now expands the planning portion of the functional model to illustrate the analytical method and indicate lower level functions within each general function. An hypothesis implicit here is that knowledge-based systems might offer appropriate technical solutions to the problems of decision support for aircrews. This type of analysis is intended to support the investigation of that hypothesis by exposing the essential functional requirements for the man-machine work system incorporated in an aircraft in performing a military mission.

Aircraft missions represent the lowest and most "active" level of the military Air Operations hierarchy. Aircraft and their crews are the ultimate actors in the conceptual chain from commander's desire to military effect, and as such are most directly involved in combat actions. Yet, due to the pace of air combat, the range and scope of operations and the potential impact of payloads, considerable initiative and difficult decisions are reserved for individual combatants at this level. Aircrews must blend and combine the military objectives of the mission with the immediate demands of safe and effective flight in a hostile environment.

Modern sensors and weapons have dramatically increased the operational potential and thus responsibilities of individual aircrews. Crew-aiding systems, that have previously focused on enhancing human skills (aircraft flight control or weapon aiming), do not provide adequate support for the human cognitive processes required for current and future military aircraft and their attendant missions. As indicated in a previous AGARD report [2.3], these modern conditions now emphasize the "managerial" - cognitive and decision-making - workload of the aircrew. For that reason, on-board decision support systems for aircrews have been the subject of much study and discussion in recent years.

The "work-system" is here considered to be the combination of human crew and machines assigned to complete the mission objective. In general, this system consists of the crew, pilot-vehicle-interface and associated decision support systems. The "system-to-be-controlled" is the aircraft. In general, it consists of aircraft, flight systems, sensor systems, weapons systems, decision support systems, navigation systems, crew systems, and communications systems.

Pre-mission plans, developed by aircrews and/or planning systems exist as partial plans for implementation. When problems are diagnosed, solutions might be found in several general forms. First, existing partial plans might have been prepared for the class of problems that have been identified. For example, in an aircraft on a Combat Air Patrol mission, the Diagnosis of a radar track as an enemy fighter aircraft approaching at high speed is an anticipated problem. In such a case the Coordination function directs the problem to Plan Selection where the appropriate plan is selected and specified to the problem. Thus, the solution to the problem is found in a choice between prepared attack plans - perhaps front quarter long range, beam medium range or stern conversion to short range - or an avoidance plan, depending on the details of the situation. Figure 2.18 illustrates the process.

The process shown in Figure 2.18, begins with a search for appropriate plans among available candidate plans. Thus, in the example, an imperative to defend certain airspace would eliminate "avoid combat" plans from consideration, and the lack of forward quarter ordnance would further reduce the search space to attacks from the beam or stern. Preliminary selection is then followed by specification of the candidates to the actual situation. That is, the particular details of precise range, speed, altitude, etc. would be entered into the partial plan. This then allows the plan to be projected or simulated and the results evaluated. Thus, a beam attack trajectory could be calculated and found to be possible, a rear quarter trajectory barely possible and a stern conversion impossible, thus eliminating that option from the selection. The remaining candidates are then evaluated, and the best is selected. In the example, the greater likelihood of a kill associated with the rear quarter attack might earn preference over the easier but less effective beam attack. Throughout the process, knowledge of mission objectives and importance, acceptable risk, enemy capabilities and tactics, possible support and a number of other details guide the final selection.

In other cases, a problem for which no previously prepared plan exists might arise. In this case, the solution might be determined to be beyond the actual capability of the work system and Coordination will request assistance, either in the form of support or relief from the requirement form external agencies. Or, the work system might determine that either it is capable of reaching a solution, or that no assistance is available. In this case, Coordination directs the problem to a function labeled Plan Generation to attempt the formation of a new plan to meet the situation. Figure 2.19 is an expansion of that function.

In Plan Generation, an attempt is made to formulate a plan to respond to an unanticipated situation. The precise process of formulating a new plan, or even some effective abstraction of that process, is the subject of considerable study and discussion. It is, in fact , a major field of endeavor in the realm of knowledge-based systems and a number of techniques have emerged to address this topic. Figure 2.19 indicates a single functional perspective for the primary purpose of stimulating thought and exposing some aspects of generative planning.

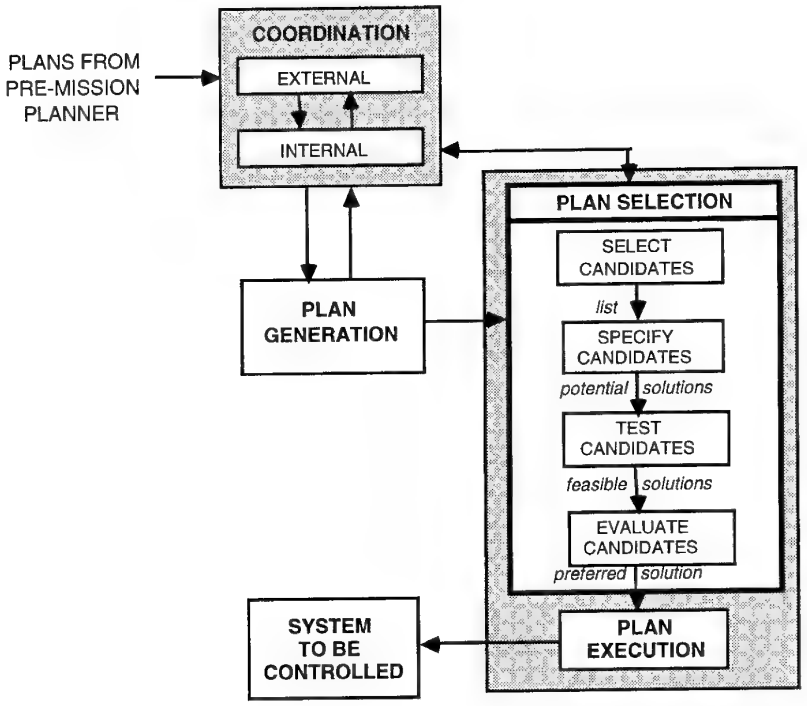


Figure 2.18 Further Decomposition of the Plan Selection Function

In this perspective, a strategy is selected based on some particular desired effect. Thus, in the case of a ground attack, aircraft suddenly called upon to perform an air-air role, such as disrupting an enemy air attack, a strategy stressing surprise or confusion might lead to the consideration of tactics intended to result in confusion. The strategy thus constrains the search for tactics, which are seen as partial plans focused on resources that may be brought to bear on the problem. Tactics further specify the kinds of plans that may be developed.

At this point, the high level goal or specified objective is decomposed into a list of sub-goals or intermediate states that lead to the desired final state. These are then sequenced or prioritized and matched to appropriate actions intended to produce the sub-goals to form a candidate plan. This plan is tested for feasibility. If it proves to be infeasible, then alternate resources or goal decompositions will be tried. If feasible, the plan is tested for effectiveness. In urgent situations, a simple threshold effectiveness value might indicate selection of the plan. If time permits, alternate plans may be generated and passed to Plan Selection where the most effective will be selected at execution time.

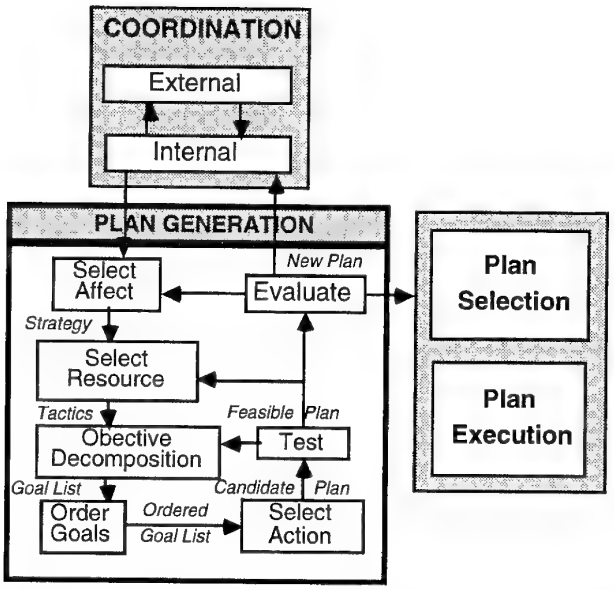


Figure 2.19 Further Decomposition of the Plan Generation Function

As previously mentioned, the Plan Selection Function comes into play when an opportunity arises to select among available candidate plans. In general, this permits a more detailed specification of partial plans and evaluation of multiple candidates. If time permits, the process can be driven down to the level of actions and operators to result in a highly specialized plan. This is expected to result in the selection of a better plan for execution.

Plan Execution is largely a reactive planning process. That is, Plan Execution is a function that closely monitors events and effects at a fine-grained level and tunes or refines the action list of an executing plan. Thus, the Plan Execution Function is charged with selecting specific operators in the normal execution of a plan, of noting minor changes in the world state that warrant slight alterations at the sub-goal level and result in a plan alteration, and finally in conjunction with the monitoring function, for recognizing that changes in the world call for re-planning. This function also coordinates and arbitrates conflicts between concurrently executing plans and identifies local opportunities. Finally, this function issues commands to the System-to-be-Controlled that result in the system behavior intended to alter the world state.

2.2.2.6 Potential Applications of Knowledge-Based Systems to Military Air Operations

Knowledge, in the form of experience, judgments, and rules, drives these aircraft oriented work-systems. Tools, intended to improve the performance of the work-systems ought to account for and incorporate the relevant knowledge. New tools will demand the development of a new body of knowledge. In particular, tools developed to support decisions about and management of complex systems in a difficult environment, appear to benefit greatly from a knowledge-based approach. Such tools, if developed at a highly abstracted, or "data," level require their users to learn much about them and tend to focus the user's attention on the tool rather than the problem it is intended to solve. Such tools run the real risk of complicating and possibly degrading rather than improving the work-system. Tools developed at a more robust "knowledge" level should prove adaptable to a greater range of operational conditions - including the user's actual intent - and thus direct the user's focus on the problem. Thus, knowledge-based tools appear to offer great potential for genuine improvements in the work-system.

To further clarify the distinction between "data level" tools and Knowledge-Based tools. Consider, for example, the central problem of air-to-air combat: that of shooting down an enemy fighter aircraft. In earlier times, data level tools were provided to address the problem "calculate aimpoint for a target at range R." This problem can be solved by straightforward ballistic calculations and is highly reliable when the human crew member can maneuver the aircraft into the assumed firing position at the proper range. A tool intended to solve the "show me how to hit the target" problem implies a system that has knowledge of (a) the maneuvering qualities of the launch platform, (b) the maneuvering qualities and destructive capability of the weapon, (c) the capability and predicted potential for the target to defeat the weapon and (d) the minimum probability of kill the human is willing to accept. For the next

higher level - which may be seen in requirements for next generation fighter aircraft - the problem "plan an attack to hit the target," merits a task description that is a lengthy and highly conditional discourse. The process for developing such a task description is knowledge acquisition, and the proper application of the resulting knowledge will be embodied in a knowledge-based system.

A wide range of problems encountered by modern military aircraft fall into this category and thus the potential for application of knowledge-base techniques to their solutions appears to be extensive. Table 2.2 is a general list of such problems or Operational Requirements that seem to be amenable to knowledge-based solutions. It is believed that a wide range of Knowledge-based techniques described in greater detail in Chapter 6 may be fruitfully brought to bear on any of these problems, with the greatest potential for near term application listed in the column entitled "High Value KB Technologies." Associated issues and requirements for improvements in supporting technologies are seen in the "Technical Infrastructure Requirements."

2.2.3 Air Traffic Control

2.2.3.1 Introduction

The objective of Air Traffic Control / Air Traffic Management is to ensure safe, efficient and timely operations of a large number of aircraft using the same airspace at the same time[5].

A pilot of an individual aircraft generally has very little knowledge about and no control of the other traffic. Consequently an independent, ground-based authority, i.e. Air Traffic Control (ATC), has been established to coordinate and control all traffic operations in a given air space.

Air Traffic Control is a typical *Work System* where human operators make use of a variety of ground-based and on-board *Sensor Systems* to collect information. Similarly, they use different ground-based and, largely through the pilot on-board, *Effector Systems* to implement their intentions and commands. However, most, if not all, *Processing Functions* are currently still carried out in the brains of human controllers. In many high density traffic areas, both in the air and on the ground, the human control capacity has already become the limiting factor in the overall system performance of the ATC system. In order to cope with future increased air traffic demand and to overcome the limitations of the human information processing capabilities, more and more of the information processing functions of the human ATC controllers must be supported by or even replaced by intelligent machine functions.

The basic structure of the *Information Processing Functions* in *Air Traffic Control* and the potential for supporting or replacing these functions through the application of knowledge-based technologies is described in the following paragraphs.

2.2.3.2 Generic Decomposition of Information Processing Functions in Air Traffic Control

There is an extremely broad variety of functions that must be performed in Air Traffic Control. Among them, are two fundamental safety related functions: (see Figure 2.20)

- Establishing and maintaining safe separation between aircraft
- Providing flight path (or ground path) guidance information/instructions to the aircraft.

In order to fulfill these core safety functions there are many derived auxiliary functions including:

- Trajectory prediction
- Conflict detection
- Conflict resolution, etc.

A host of other functions are performed to meet the higher level objectives of improved efficiency and timeliness, e.g.:

- Airspace management
- Traffic flow management
- Sequencing and scheduling, etc.

---

5 This concept generalizes to ground-based operations as well.

Regardless of the specific functions to be performed in ATC and regardless of whether a function is fully automatic or is implemented by a human *Operator,* who either uses or makes no use of *Tools,* the generic, top-level functional structure introduced in Section 2.1 can be applied as illustrated in Figure 2.20.

At the highest level of representation, *the System-to-be-Controlled* is the total Air Traffic Services (ATS) system. At lower levels, the System-to-be-Controlled can be any other unit of the ATS-System, such as: Flow Management, Sector En-route Control, Approach Control, Ground Movement Control or even individual aircraft control. Thus, the objects under control include planned flights, ground movements, gate assignments and the actual flying aircraft. The current, state of the aircraft (e.g., aircraft position, identity, altitude, as well as flight plans) is derived from various sensing and surveillance systems, databases etc. These internal and external state signals are used as input data for *Situation Assessment.*

Situation Assessment comprises two sub-functions: *Monitoring* and *Diagnosis.*

The Monitoring function in ATC is performed continuously and recurrently. Aircraft identity, position and altitude are collected. Heading, track, average ground speed, rate of climb/descent, etc. can be calculated. These actual state values are permanently observed and checked to determined whether they match with the desired state values of a flight. If there are *no abnormal deviations,* the system may continue with the current plan, whereby the necessary *Commands, Objectives* and *Constraints*, (e.g. "descend to Flight Level 230"; "cleared ILS approach 25R", etc.) will be transmitted for actuation to the system-to-be-controlled.

If, however, the observed state *deviates from the expected* state, the *Diagnosis* function is activated. Here the state of an individual planning object (an aircraft) as well as the overall traffic situation is assessed, analyzed and evaluated. If significant deviations (e.g., an altitude deviation) have been detected, the actual state/situation has to be compared with the desired state/situation, which is given by either an initial plan or by the current plan. To carry out the *Diagnosis* function, a variety of prediction, extrapolation, filtering and evaluation techniques may be applied along with stored models, rules, strategies, criteria, objectives, and constraints, which may have been imposed by either an *internal Coordination* or by a higher level *external Coordination* through the internal coordination. If the deviation matches all required constraints and conditions and also does not affect other traffic, the *Diagnosis* is finished and the Monitoring function may be continued.

If, however, a problem (e.g., a flight path conflict with another aircraft) or an opportunity (e.g., improved actual weather in comparison to predicted weather) has been detected, the *Plan Generation* function is activated. Its task is to resolve the problem and take advantage of the opportunity that has been detected. To *generate a plan* means to find a solution and a set of actions that transform the current state into a *tentative, desired* future state or that achieve a solution that optimizes a stated criterion (e.g., maximum landing rate) subject to stated constraints (for example, safety or operator workload). In either case, the generated plan results in system state wherein the problem no longer exists. Due to the nature of a problem, it may be necessary to decompose the problem into several sub-problems and to resolve these consecutively. It may be possible or desirable that the *plan generation* function develop not only one potential solution but several alternative solutions, all being able to resolve the problem or optimize the situation in different ways.

In the standard case, the *Plan Generation* function applies pre-stored, standard models, rules, algorithms, human heuristics or experience, etc. for problem solving under consideration of typical, standard objectives and constraints. These objectives and constraints may however be modified or overruled by others being imposed by *internal/external coordination* to deal with unforeseen, exceptional events.

As Air Traffic Control is not a monolithic, stand-alone work system but is composed of a great number of both parallel and hierarchical interconnected sub-systems, the solutions developed by the *Plan Generation* function of one sub-system has the potential to interfere with or affect other *Plan Generation* agents of other sub-systems. Thus, a planned solution often must be coordinated with other external agents.

Thus, depending on the special circumstances, the *Coordination* process may be a mere "tell-and-listen" dialogue or a more intense negotiation process, which is an intelligent function in itself, requiring its own methods and techniques. The outcome of the *Coordination* with other *internal* or *external* agents (e.g., supervisors in other control sectors, in flow management units, etc.) either confirms the feasibility of the potential solution(s) or triggers a new *Plan Generation* cycle until a solution is found which satisfies all conditions and constraints.

Once a plan or different options have been generated, they must be transferred for *Implementation.* If there are several options, the *Plan Selection* function chooses which of the potential alternative solutions or plans is to be

applied. According either to given pre-stored goal criteria or to objectives, constraints or strategies imposed by *internal/external Coordination,* the different solutions are evaluated and compared, and the best solution/plan is selected.

Table 2.2 Potential Applications of Knowledge-Based Systems Technology

Operational Requirement	**High Value KB Technologies**	**Technical Infrastructure Rqmts.**
Mission Panning	Prob Solvg, ExpSys, McLrn,	KnowEng, RlTm, DtBs, N/W
Tactics Planning	Prob Solvg, ExpSys, IntFc	KnowEng, RlTm, DspSys
Emergency Planning	Prob Solvg, ExpSys, IntFc	RlTm, DspSys
Divert Planning	Prob Solvg, ExpSys, IntFc	RlTm, DtBs, H/W, N/W,DspSys
Maintenance Planning & Scheduling	Prob Solvg, ExpSys, McLrn	KnowEng, DtBs, H/W, N/W
Flight Systems Mgmt.	Prob Solvg, ExpSys, IntSes	RlTm, H/W
Sensor Mgmt.	Prob Solvg, ExpSys, IntFc	KnowEng, RlTm, DspSys
Multi-sensor Data Fusion	ExpSys, McLrn, IntSes	KnowEng, RlTm, H/W, N/W
Automatic Target Recognition	ExpSys, McLrn, IntSes, IntFc	KnowEng, RlTm, DspSys
Communications Mgmt.	ExpSys, McLrn, IntSes, IntFc	KnowEng, RlTm, N/W
Situation Assessment	Prob Solvg, ExpSys, IntSes, IntFc	KnowEng, RlTm, DtBs, DspSys
Electronic Warfare Systems Mgmt.	Prob Solvg, ExpSys, McLrn, IntSes	KnowEng, RlTm, DtBs, H/W, N/W
Reconnaissance Systems Mgmt.	Prob Solvg, ExpSys,IntSes	KnowEng, RlTm, DtBs, N/W
Surveillance Systems Mgmt	Prob Solvg, ExpSys, McLrn, IntSes, IntFc	KnowEng, RlTm, DtBs, N/W, DspSys
Defensive Systems Mgmt.	Prob Solvg, ExpSys, IntSes, IntFc	KnowEng, RlTm, N/W
Cargo Mgmt.	ExpSys, McLrn, IntSes	KnowEng, DtBs, N/W
Battle Group Mgmt.	Prob Solvg, ExpSys, IntFc	KnowEng, RlTm, DtBs, N/W, DspSys
G&C of Lethal UAV	Prob Solvg, McLrn, IntSes,	KnowEng, RlTm,H/W
G&C of Non-Lethal UAV	Prob Solvg, McLrn, IntSes	KnowEng, RlTm,H/W, N/W
LEGEND:	Knowledge Engineering applies to all Prob Solvg=Problem Solving&Search ExpSys = Reasoning about Physical Systems McLrn = Machine Learning IntSes = Intelligent Sensing; Understanding External Env IntFc = Human-Machine Interaction	KnowEng = Knowledge Engineering RlTm Real Time performance H/W = High Performance H/W N/W = High Performance Network

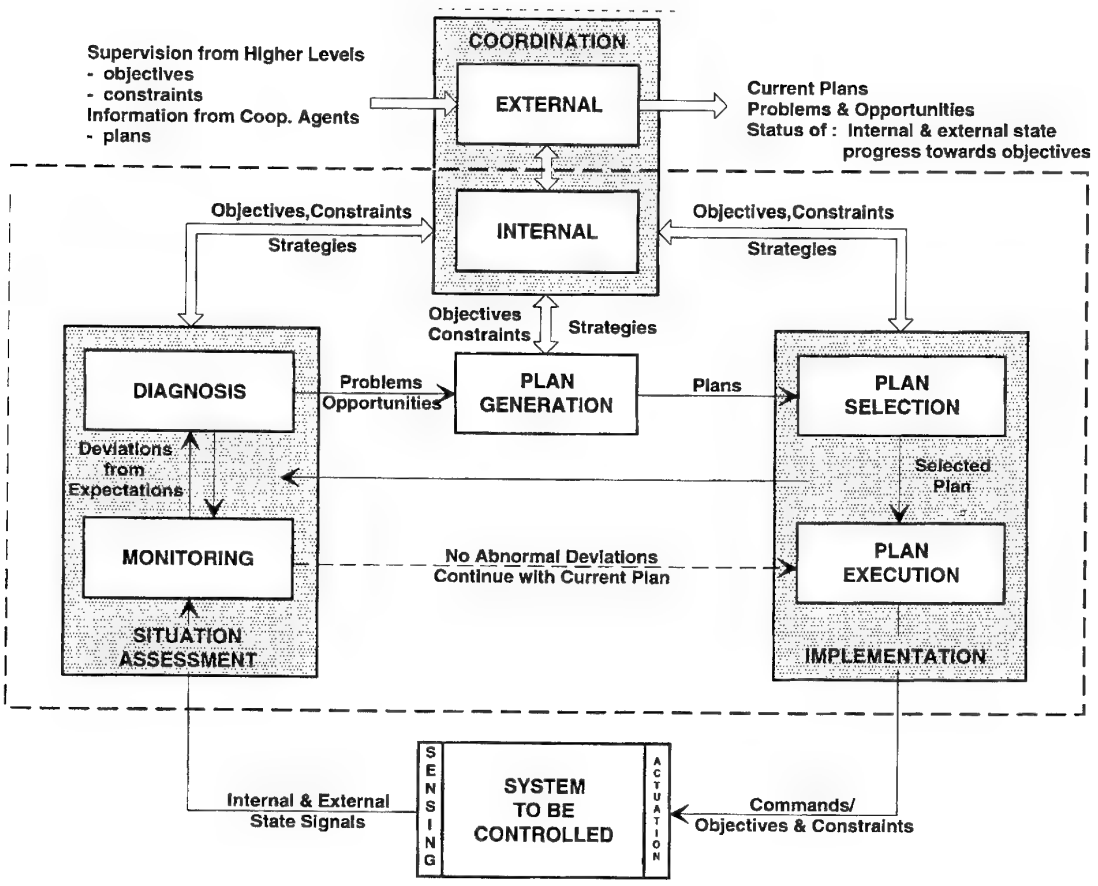


Figure 2.20 Generic Structure of Decision-Making / Problem Solving Subfunctions

In a final step, the selected plan is executed. *Plan Execution* translates the solution/plan into a sequence of actions (e.g., heading, speed or altitude changes) to achieve the desired state, e.g., to avoid a conflict with other aircraft.

These actions are sent as commands/advisories to the *Actuation* element of the *System-to-be-controlled*. In the current ATC-system, commands usually are sent to the pilot via VHF communication and the pilot then provides them as input commands to the aircraft's actuation systems. In the future, a digital data-link will provide the potential for direct digital communication/cooperation between ground-based and on-board automated systems.

The generic information processing structure described above is generally valid for all information processing functions in ATC. Its applicability ranges from the very broad, high level strategic planning functions many months in advance of an actual flight, down to the very specific sub-tasks, e.g., resolving within seconds short term conflicts between two aircraft. Depending on the hierarchical level and the magnitude and complexity of the tasks to be performed, these functions may be carried out either:

- in the brain of a human operator
- by intelligent machines to support and enhance the human skills
- by machines only.

2.2.3.3 Decomposition of ATM-Functions

Air Traffic Control (ATC) is part of an overall Air Traffic Management (ATM) system. Advanced operational concepts for ATM/ATC are presently under development in North America (AAS [2.16]) and in Europe (EATMS [2.17]; CATMAC [2.18]). They follow a well designed, consistent architecture in which all ATM/ATC functions are deliberately interconnected and performed by several autonomous, but cooperating planning agents. For example, in the German CATMAC (Cooperative Air Traffic Management Concept) proposal, the functions are decomposed:

(1) In terms of time:

strategic/long-term ---> tactical/medium-term ---> short-term planning ---> actual control

(2) In terms of space:

global ---> continental ---> regional ---> local

In particular, in the German CATMAC the four time horizons are considered as four hierarchical levels of one consistent concept. Two of these levels namely: Air Traffic Flow Management (ATFM) and actual Air Traffic Services (ATS) are illustrated in Figure 2.21 as an example.

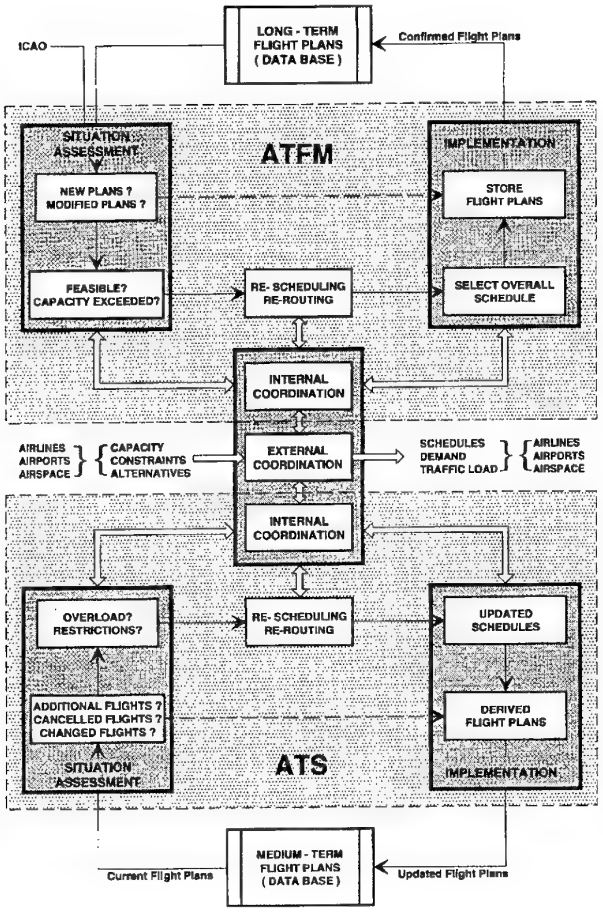


Figure 2.21 Air Traffic Flow Management (ATFM) and Air Traffic Services (ATS)

The System-to-be-Controlled by ATFM at the "highest", long-term, strategic level does not comprise real, flying air traffic, but *planned* air traffic, i.e., flight plans which are stored on paper or in electronic data-bases. The internal and external state signals are flight plan requests which are filed by the airlines to ICAO through organized flight plan conferences usually up to one year in advance. Other external state signals include the expected future air and ground capacities of the Air Traffic system. First Situation *Assessment* is performed. In the *Monitoring* function these new or up-dated flight plans are checked against already existing plans. If deviations from expectations are detected, i.e., more requested flights, major changes in the schedules etc., the *Diagnosis* function will have to find out if these changes still comply with the objectives, constraints etc. imposed by internal resources (e.g. ATM/ATC capacity) or by external agents (e.g., the Airport Authorities, Airlines, military restrictions, night curfews, etc.). If problems are detected, i.e., requested flights exceed available capacity or flight schedules at restricted times etc. *Plan Generation* is activated to find solutions and to establish a smooth traffic schedule that meets all requirements. If necessary, the potential solutions are coordinated with ATM/ATC, with the Airlines and with the Airports through *internal* or *external Coordination*. The generated plans are prepared for *Implementation.*

The final plan is selected according to the objectives, constraints etc. passed along from other agents through *Coordination. The Implementation* of the flight plans is accomplished by distributing the agreed flight schedules to all participating parties. This long term plan finally becomes the initial plan for the next level of planning: ATS / ATC.

At the next ATM level, ATS-Flight Planning, the medium-term flight plans represent the basic *System-to-be-Controlled.* In the *Situation Assessment* function, the external (ATFM flight plans) and the internal (currently activated flight plans) are continuously monitored.

If there are major deviations, i.e., additional flight plans yet to be approved, canceled flight plans, etc., the *Diagnosis* function must check for problems or opportunities with respect to the objectives, constraints, etc., that have been imposed through *internal or external Coordination* (i.e., flow restrictions, slots, temporary usage of military airspace, etc.). The solutions generated by *Plan Generation* are then transferred to the Flight Plan Processing system for *Implementation.* All unchanged standard flight plans will be activated automatically some hours before the flight actually commences. All other changes will be added or deleted, giving a complete, updated schedule for all flights to be conducted within the next hours.

These plans may still become updated if necessary. Actual excerpts from this comprehensive medium-term plan for the following short-term planning period (~ 60 min) are extracted and are sent through the *Coordination* function to the next level of ATM, so-called short-term planning, where the plans are used as initial plans and will be processed in an analogous cycle of *Monitoring, Diagnosis, Plan Generation, Plan Selection and Plan Execution.* This level and the subsequent Control level of ATM/ATC are not shown in Figure 2.21.

2.2.3.4 Knowledge-based Functions in ATC - Arrival Planning

One of the first applications where an ATC planning function has been performed with the assistance of knowledge-based technology has been *Arrival Planning, Sequencing and Scheduling.* A first operational installation has been put into place in 1989 at the Frankfurt, Germany ATC-Center. The system, named *COMPAS* (Computer Oriented Metering Planning and Advisory System) is described in more detail in Chapter 6.1.4.

Figure 2.22 illustrates the Arrival Sequencing and Scheduling functions as part of the *short-term-planning layer of ATM/ATC.* At this level, the *System-to be controlled* is the *actual Air Traffic*, a set of inbound aircraft. Their flight plans, positions, altitudes and identifications are continuously sent from different sensor systems to the *Situation Assessment* function. Here, headings, tracks, speeds, descent profiles etc. are continuously calculated and sent to the *Diagnosis* function. The *Diagnosis* function predicts, extrapolates and correlates future trajectories to detect deviations from the plan and to detect potential future conflicts. If a planning conflict has been found, the *Plan Generation* function is activated. It attempts to resolve the conflict by using stored solutions or stored problem solving methods. The *Plan Generation* results represent tentative solutions for the Sequence, Schedule and Trajectories for the inbound flights. These planned and still tentative solutions must be coordinated with other planning agents (e.g., adjacent upstream ATC-sectors, the downstream Tower Sector, with Departure Control). After an agreement has been reached through *Coordination*, the potential solutions are transferred for *Implementation*. Here they are evaluated and the "best" solution with respect to a given goal criterion is selected. The solution is executed by the *Plan Execution* function which transforms the solution (Sequence, Schedule, Trajectory) into commands (Heading, Speed, Descent, Intercept etc.) which are transmitted to the Systems-to-be-Controlled: the arriving aircraft.

Some of these functions can have the potential for being implemented by an intelligent computerized planning system to support the human *Operator* in the control of *Arrival Traffic*. In *COMPAS*, the *Monitoring, Diagnosis,* and *Planning* functions are performed automatically and continuously by the computer. The results, the *COMPAS Plan*, are presented through a specially designed Human-Computer-Interface (HMI) to the human *Operator*. The human Controller can integrate the *COMPAS* generated plan into his other control activities and retains the ultimate authority for decision making and implementation. He is also able to interact with the computerized *Planning* function through the HMI. Figure 2.23 shows the cooperation between human and computer-based functions for the *COMPAS* system.

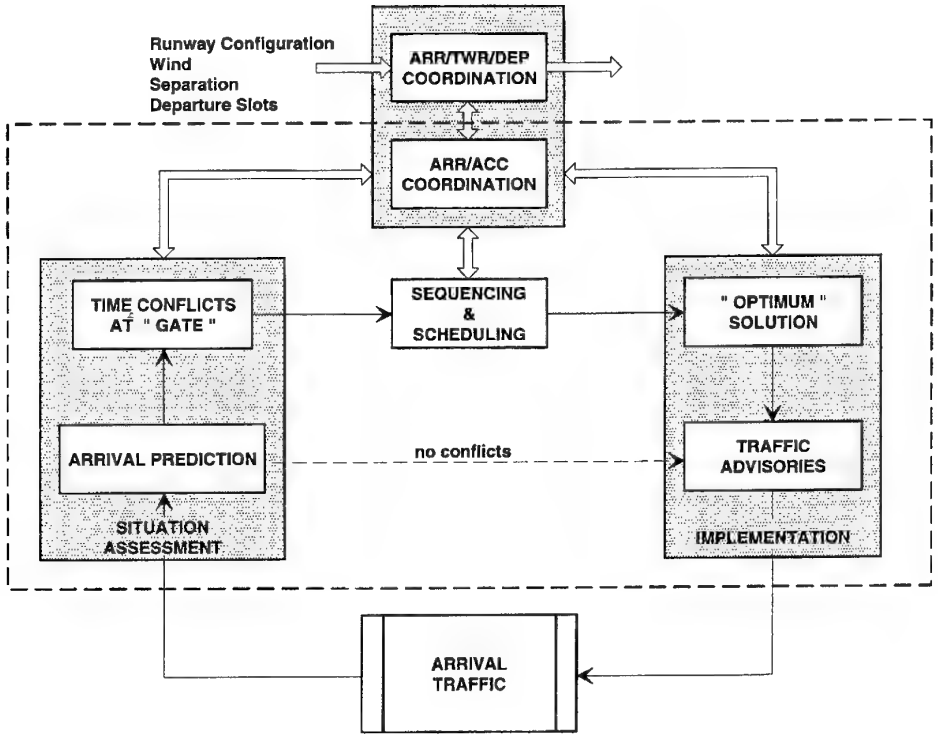


Figure 2.22 Arrival Sequencing and Scheduling Functions

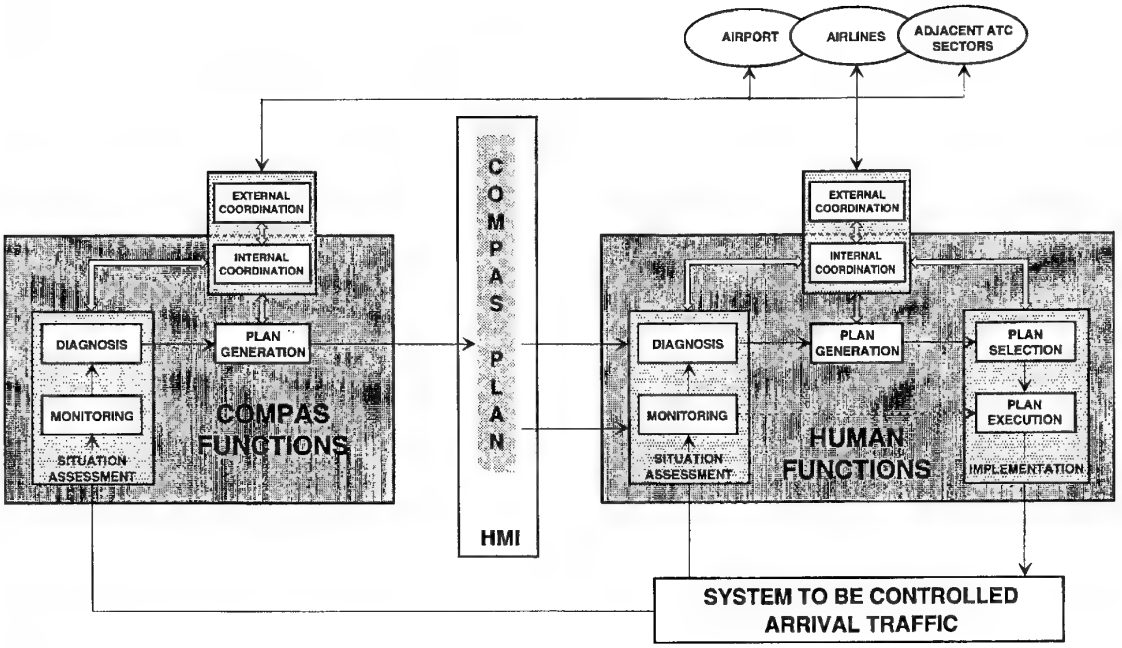


Figure 2.23 Human Machine Interactions in COMPAS

2.2.3.5 Challenges and Problem Areas with a Potential for KBS Applications in ATC

Four characteristics of Air Traffic Control (ATC) problems pose major challenges:

- Very Large Scale System
- Problem Complexity
- Solution Flexibility, Robustness
- Human-in-the-Loop

Each of these challenges is addressed briefly in the following. paragraphs.

Very Large Scale System

As shown in Section 2.2.3.2, ATC is a very large scale system in terms of both time and space. The temporal scope is from long-term to the very immediate short-term. The spatial scope is from global to local. A broad variety of functions of different detail and character must be carried out in parallel at different levels, distributed over different time horizons and at different locations. Still, all information processing is interrelated and has to be interconnected in numerous control loops. There is an urgent need and a vast potential for the application of advanced knowledge-based functions for:

- Hierarchical planning
- Multi-Agent planning
- Multi-Sensor-Data-Fusion
- Information Fusion
- Information Management
- Filtering

Problem Complexity

Most of the planning and control functions in ATC are highly complex. Many different requirements and constraints originating from Airport operators, from Airline operators, from pilots and from the environment (noise, pollution avoidance), which very often have competing or even contradicting goals, must be considered simultaneously. Some functions must be performed cooperatively between on-board and ground-based systems. Other ground-based functions and tasks are divided and allocated or shared among several different ground units.

These challenges call for new applications of knowledge-based solutions for:

- Multi-Agent Planning
- Consistency and Completeness
- Advanced Planning Technology

Solution Flexibility / Robustness

Despite the application of the most advanced sensor technologies and data processing capabilities in ATC, it remains a significant challenge for planning and control functions to adapt continuously to changing conditions, i.e., to close all loops in real time. As ATC is, in principle, largely a customer service system, it must comply with Airline, Pilot, Passenger, and Airport needs. Unforeseen events, disturbances, and changing priorities are commonplace and occur on short-notice. Weather (headwinds, fog, thunderstorms, etc.) frequently add to the problems of uncertainty in ATC-Planning and Control. Consequently, some processes are of a rather non-deterministic nature. New developments in

- Planning with Uncertainty
- Qualitative Reasoning
- Heuristic Planning
- Fuzzy Logic

offer the potential to meet these challenges.

Human-in-the-Loop

Despite all of the advanced technology either on-board or on the ground and despite the expected increasing number of intelligent implementations of functions in ATC, it is unlikely that in the foreseeable future there will be aircraft flying automatically without a pilot onboard, in airspace being automatically controlled without controllers on the ground. Thus, there will still be pilots and controllers in charge and responsible for the conduct of air traffic.

Human limits in perception and information processing and the typical human approaches to planning and decision making (heuristic, wholistic) all impose very interesting but severe challenges on the designers of Planning and Decision Support Systems, who must model and transfer human cognitive processes to intelligent machines.

Only when both the representation of information and the manner of dialogue and interaction with an intelligent device are acceptable to the human operator, will knowledge-based functions be successfully implemented, no matter how intelligent and advanced they may be. New knowledge-based functions such as:

- Heuristic Planning
- Real-Time-Expert-Systems
- HMI-Interaction Techniques
  - Speech Understanding
  - Natural Language, HyperMedia
  - Virtual Reality
  - Adaptive Interfaces, etc.

offer the potential to meet these challenges.

2.2.4 Functional Analysis Of Spacecraft Mission Operations [2.19, 2.20]

2.2.4.1 Introduction

This section outlines a baseline structure for knowledge-based guidance and control functions in the context of space missions. Guidance and control functions will be referred to here as "Spacecraft Mission Operations", i.e., all the functions required to implement space missions. The development of a general baseline structure is difficult since these functions may vary considerably from manned space mission to unmanned mission, or from an earth observation satellite mission to a communications satellite mission.

This section focuses on unmanned missions since they correspond to a majority of the actual space missions, and it presents a generic structure that is applicable to any kind of unmanned mission. Adaptation of this structure to manned space missions (e.g., Space Shuttles, Space Stations) is also discussed.

2.2.4.2 Functional Structure Of Spacecraft Mission Operations

A spacecraft is generally composed of two main parts :

The payload	composed of any on-board equipment directly associated with the mission itself including: observation instruments and associated electronics for remote sensing satellites, antennas and communications electronics for communications satellites, telescopes and associated electronics for scientific satellites
The platform or bus	composed of any service subsystems required for supporting the general spacecraft operations including: solar arrays, attitude and orbit control, propulsion, thermal control, structure, telemetry, telecommand, on-board data management

Spacecraft mission operations generally consist in managing the spacecraft : more specifically, its two main parts, the payload and the platform for meeting given mission goals (e.g., accommodating customer's requests or scientific objectives).

Unmanned spacecraft are automatic systems that are teleoperated from the ground. These satellites operate at varying degrees of autonomy with the level of autonomy depending on several factors :

Type of orbit for earth observation satellites stationed in low earth orbits (altitudes of 600 - 800 Km) the visibility from the ground control center is limited to typically 10% of the mission duration, thus requiring some automatic operations on-board the satellite during the 90% of the orbit when control communications are interrupted

Type of mission military communications satellites are far more autonomous than civilian ones

Economic constraints commercial space communications organizations such as INTELSAT, INMARSAT or EUTELSAT are committed to reducing their operations costs, and consequently, there is motivation to automate the mission operations

Considering the current state of the art, whatever the level of autonomy, most of the mission operations tasks are still performed on the ground. These tasks are, if we take the example of an earth observation satellite system concentrated in two main ground entities :

- Mission Control Center (MCC)
- Spacecraft Control Center (SCC)

The first entity, the Mission Control Center, is primary dedicated to Mission Planning, i.e., generating mission plans to be executed by the spacecraft through telecommands sent by the Spacecraft Control Center. Mission plans are generated in accordance with mission goals or customer's requests and in consideration of general mission conditions (e.g., spacecraft orbital position, seasonal conditions), the current state of the spacecraft (provided by the SCC to the MCC on the basis of telemetry analysis), and, depending on the type of mission, the mission data reception (e.g., images for an earth observation satellite).

The second entity, the Spacecraft Control Center, is responsible for controlling the mission execution through telemetry parameter analyses. These parameters are regularly downlinked by the spacecraft. The SCC also commands and monitors the mission execution, according to the mission plans provided by the MCC and on the basis of predefined procedures. Thus, the main SCC functions are telemetry and telecommand processing.

Additional functions of the SCC :

Flight Dynamics compute spacecraft orbital positions, and thus determine appropriate orbital maneuver strategies for orbital corrections

Specific Software Packages perform specific monitoring tasks, e.g., on-board electrical power balance between consumption and generation

The architecture of the system described above is depicted in Figure 2.24.

All of the functions described above are performed by any spacecraft ground system, but they may be organized in different ways from one system to another. If we focus on Guidance and Control related functions, and if we refer to Section 2.1, the functional analysis corresponding to such a system can be represented as that shown in Figure 2.25 :

These functions may be performed on-ground (for most of the current space systems) or shared between on-board systems and a ground segment (for unmanned spacecraft with a high level of autonomy, or for manned spacecraft such as space shuttles or space stations.)

These main components of any spacecraft operations and are all candidates for partial or total automation, and thus, have the potential to benefit from applications of knowledge-based systems.

2.2.4.3 Problem Areas In Spacecraft Operations With Potential For Knowledge-Based Solutions

Some major trends in spacecraft operations provide strong motivation for the use of knowledge-based systems (KBS) ; they are summarized, together with the potential benefits of using KBS, for the respective spacecraft operations functions, in Table 2.3;

2.2.4.4 Concluding Remarks

The desire for increased autonomy and automation was spearheaded by the major international players in the space business (agencies, major contractors, etc.) during the 80's and has led to the first steps toward the

development of knowledge-based systems for supporting spacecraft operations. Some of these systems are now operational, or are close to being operational, and span all the major spacecraft operation domains :

- Mission plan generation (e.g., Plan ERS at ESA/ESTEC)
- Monitoring (e.g., ARIANEXPERT at ARIANESPACE and RTDS at NASA JSC)
- Diagnosis (e.g., SHARP at JPL and DIAMS at CNES)
- Mission plan execution (e.g., EOA at ESA/ESOC and PRS at NASA)
- Procedure generation (e.g., PREVISE at ESA/ESTEC and PROCSAT at CNES)

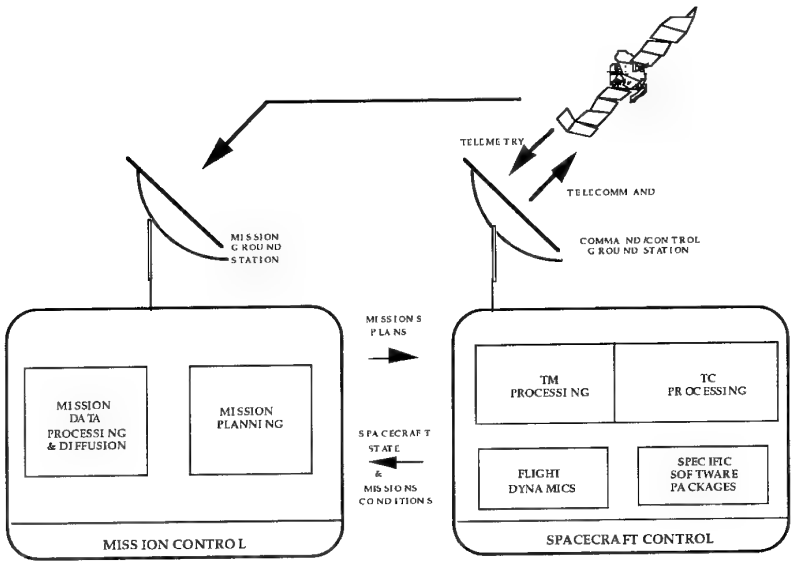


Figure 2.24 General Architecture Of A Spacecraft Ground Segment

Table 2.3 Potential Of Using Knowledge-Based System For Spacecraft Operations

S/C OPERATIONS FUNCTIONS / TYPICAL PROBLEM AREAS & CURRENT TRENDS IN SPACECRAFT OPERATIONS	MISSION PLAN GENERATION (SEE § 4.3)	MONITORING (SEE § 4.4)	DIAGNOSIS (SEE § 4.4)	MISSION PLAN EXECUTION (SEE § 4.7)	PROCEDURES GENERATION (SEE § 4.3)
INCREASED COMPLEXITY IN HUMAN-SYSTEM INTERACTIONS (SPACECRAFT COMPLEXITY ; RISK OF HUMAN ERRORS)		INTELLIGENT ASSISTANCE TOOLS WITH POWERFUL EXPLANATION CAPABILITY	INTELLIGENT ASSISTANCE TOOLS WITH POWERFUL EXPLANATION CAPABILITY	AUTOMATION OF SPACECRAFT OPERATIONS	ADVANCED TOOLS FOR VALIDATING OPERATIONS PROCEDURES
INCREASED COMPLEXITY IN MISSION MANAGEMENT (NUMBER OF PAYLOADS, NUMBER OF CUSTOMERS' REQUESTS, ...)	A NEW GENERATION OF MISSION PLANNING SYSTEMS BASED ON ADVANCED PROBLEM SOLVING TECHNIQUES				
PROJECTS & MISSION DURATION (PROBLEM OF EXPERTISE & EXPERIENCE CAPTURE)		KNOWLEDGE - BASED MONITORING SYSTEMS	KNOWLEDGE - BASED DIAGNOSIS SYSTEMS		
REQUIREMENTS FOR ENHANCED PRODUCTIVITY (e.g. TASKS AUTOMATION)	SEMI AUTOMATION OF MISSION PLANNING	AUTOMATION OF HIGH LEVEL MONITORING TASKS		AUTOMATION OF SPACECRAFT OPERATIONS	ADVANCED TOOLS FOR GENERATING OPERATIONS PROCEDURES
GENERATION OF HUGE AMOUNT OF DATA BY SPACE MISSIONS		MACHINE LEARNING FOR AUTOMATING DATA ANALYSIS			
REQUIREMENTS FOR MORE FLEXIBILITY	INCORPORATION OF DECLARATIVE ART OF PROGRAMMING IN MISSION PLANNING SYSTEMS (RULES, OBJECTS) FOR ADAPTATION TO CHANCES (MISSION STRATEGY, CUSTOMER'S REQUESTS, ...)				

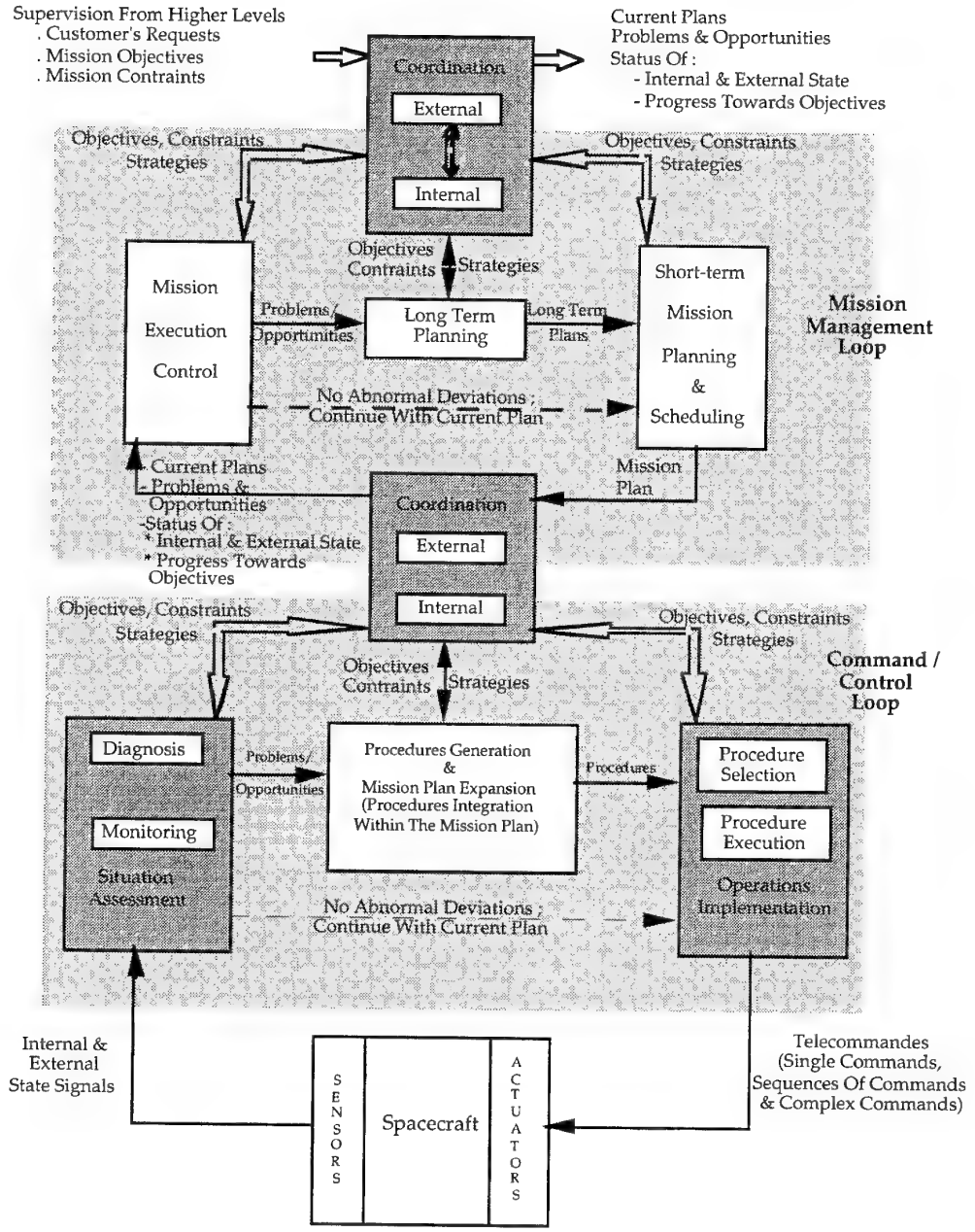


Figure 2.25 Functional Structure of Spacecraft Mission Operations

2.3 Life Cycle Management of Guidance and Control Systems

2.3.1 Introduction

In any system, the buyer's objective is to obtain the most capable system for that meets requirements for the least cost possible. Often the cost of the system is viewed as the acquisition cost, that is, what it costs to acquire the system initially. The acquisition cost ignores the additional costs associated with keeping the system operating once it is acquired. These support costs can eventually swamp the acquisition cost. Only by addressing the complete cost of the system over its life (the so-called *life-cycle cost*) can an accurate picture of the cost/benefit ratio of a system

be obtained. For example, cost balancing from the perspective of system mean time between failure (MTBF) is depicted in Figure 2.26.

Optimizing this balance is, in the ideal case, the job of an oversight function that we will term Life Cycle Management. There are three different ways in which knowledge based systems (KBS) can contribute in optimizing this total cost: 1) a KBS can aid in the oversight of the entire life cycle process, 2) a KBS can be applied to some part of the life cycle (a phase), or 3) the system whose life cycle we examine can be a KBS. We will discuss briefly the problems associated with the oversight function in Section 2.3.2. In Section 2.3.3, we examine the issues associated with each life cycle phase, and in Section 2.3.4, we discuss the life cycle characteristics of the development of a KBS .

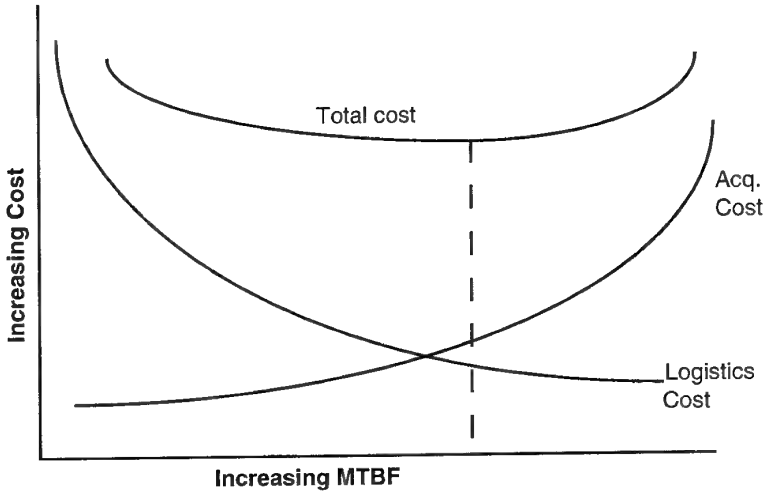


Figure 2.26 Optimizing Total Cost Contributions as a Function of MTBF

Life cycle costs can be analyzed in terms of the phases which any avionics system goes through during its term of service. One view of these phases, the so-called waterfall model, is shown in Figure 2.27. Each of these phases has a contribution to life cycle costs, discussed more fully in [2.21].

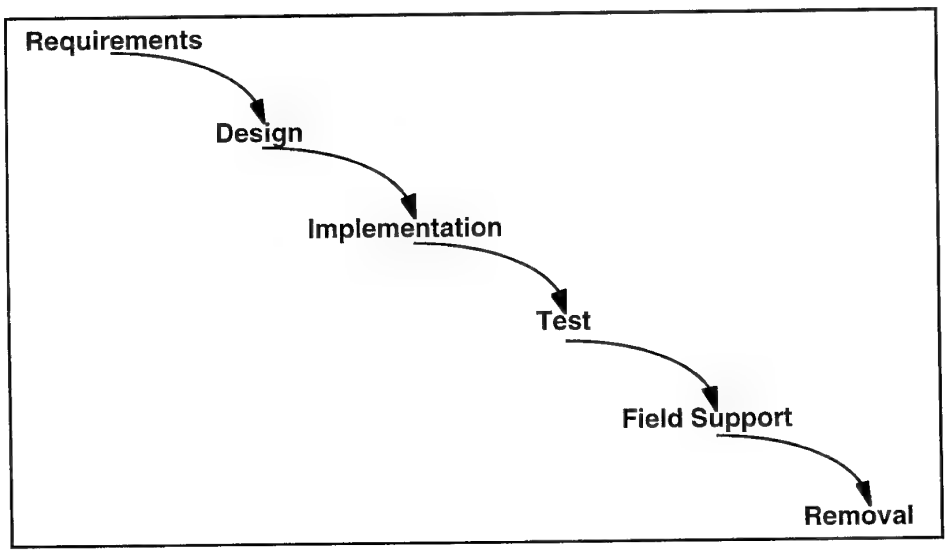


Figure 2.27 Life Cycle Phases

If the waterfall view is taken, it is clear in comparison with the functional view presented in Section 2.1 what is missing. Specifically, there are no feedback loops that enable effective control. While this feedback exists to varying degrees in existing organizations, a view which explicitly calls out this feedback allows one to analyze and optimize the feedback. One way of providing feedback within the development cycle is given by the rapid prototyping methodology, in which the requirements analysis, design, implementation and test phases are iterated, with each cycle becoming more comprehensive. This methodology is discussed more thoroughly in Chapter 5. This still omits feedback from any given phase to any other. More importantly, it omits any control function which oversees the life cycle. A more complete view which accommodates this oversight and which is more in keeping with concurrent engineering concepts is shown as Figure 2.28.

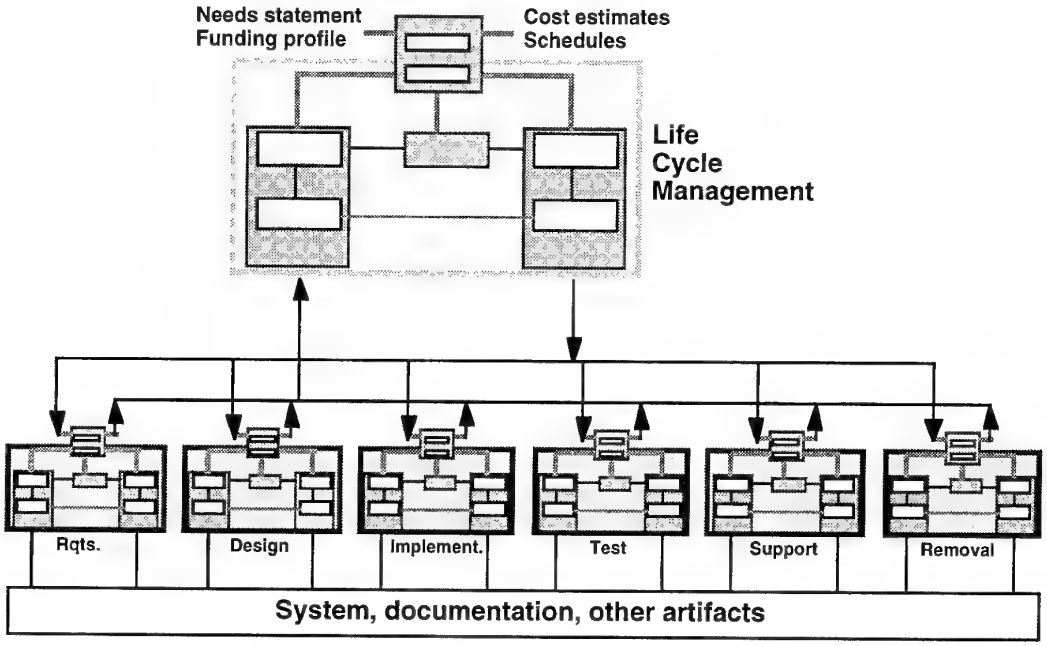


Figure 2.28 Life Cycle Management

2.3.2 Life Cycle Cost Elements

To analyze life cycle cost components, we show four major groupings of costs in Figure 2.29. These are structured in terms of the organizations that incur the costs: acquisition, operation, setting up to support the system, performing the support, and finally, retiring and removing the system from operation. Removal is a new element which is beginning to be seen as an important component of the life cycle cost and, thus, is typically not included in the breakdown analyses currently available. Life cycle cost components, excluding removal, are further refined in Figure 2.30.

Initially, there is a set of acquisition costs comprising design, fabrication, installation, and data production costs. As the avionics system is fielded, operation costs and logistic support costs come into play. Operation costs consist of personnel and other miscellaneous costs. Logistic support costs can be divided into recurring support and initial support costs. Initial support costs include spares, training and other miscellaneous costs. Recurring support costs include replacement spares, maintenance man-hours, and repair material. These contributions are shown in Figure 2.30.

Given the complexity of avionics systems being fielded today, it is difficult if not impossible for a single person to maintain a perspective which spans the life of an avionics system while remaining competent in the area that is his or her primary responsibility. Designers, for example, are often not aware that the design phase *directly* contributes only 3% to the total life cycle cost picture. Thus, producing tools whose aim is to shorten the design phase is likely to have minimal impact. This is not to say that knowledge based aids are inappropriate at this phase in the life cycle, merely that their focus should include reducing costs in other phases of the life cycle.

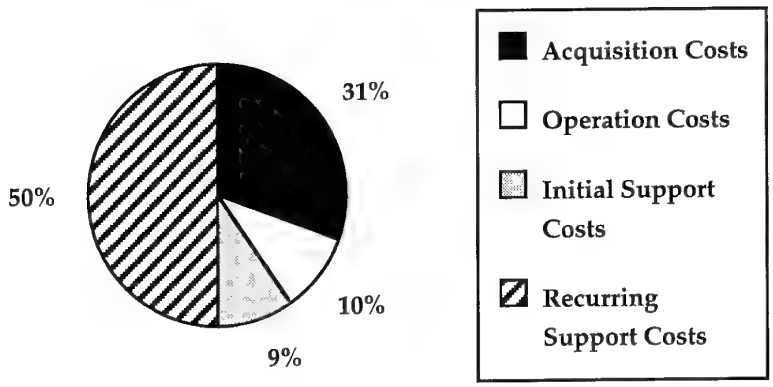


Figure 2.29 Major Life Cycle Cost Components

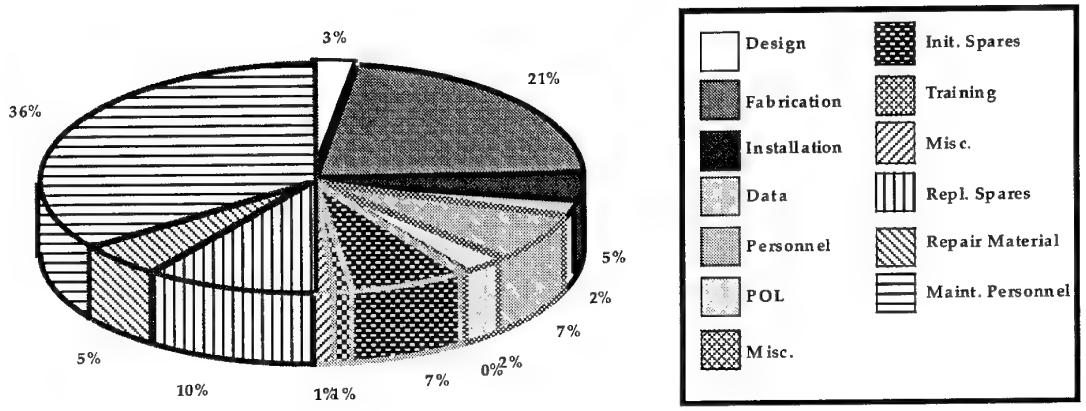


Figure 2.30 Life Cycle Cost Contributors.

Unfortunately, resources allocated to design (and all other phases) are often scarce. Obviously, designers should spend no more on design than is absolutely necessary. Diverting resources from the "real" goal to some other purpose, say testability analysis, is often viewed by design management as unproductive, even foolhardy. In order to force a wider appreciation of the goal set of the system life cycle (maximum capability for minimum life cycle cost), management focus must span the life cycle of the system. Note that this focus need not take the form of a formal organization, such as the Quality Departments set up within many companies in response to the proliferation of Total Quality Management concepts. Rather, the focus should be a knowledge of how life cycle costs are incurred and an acceptance that additional early system design and development costs should be directed toward downstream savings. In existing contracting practice, the development contract is typically separate from the later support contract, so even within the organization, there is typically no individual with the necessary span of responsibility. **Rafale** is an example of a newer contract which departs from this model, combining the support and development responsibility into a single contract and is discussed further in Chapter 6.

Recognition of the need for this management focus provides another opportunity for Knowledge Based System (KBS) introduction - a KBS which applies to and oversees the development process as a whole, rather than just an element of the process (such as design) or a capability of the system (such as trajectory optimization). An implementation of this oversight function is discussed in Section 6.3.1 in the context of the **Copilote Electronique**.

2.3.3 Affecting Life Cycle Cost

From the breakdown shown in Figure 2.28 it is clear that the primary areas in which life cycle costs might be reduced, assuming they are amenable to treatment, are in recurring support costs, particularly maintenance man-hours. The remainder of this section focuses on the nature of the problems which drive maintenance man-hours, replacement spares, and initial spares complements to be so high and on how these might be affected by the application of knowledge based system technology.

2.3.3.1 Design (Figure 2.31)

Since design occurs early in the life cycle of an avionics system, it is often targeted as an opportunity to intervene to reduce life cycle costs. Unfortunately, this is also the phase when the eventual workings of the system are least well understood. It often seems counterproductive to the designer to spend a great deal of time in analysis of, for example, testability, when the base function is not yet well defined.

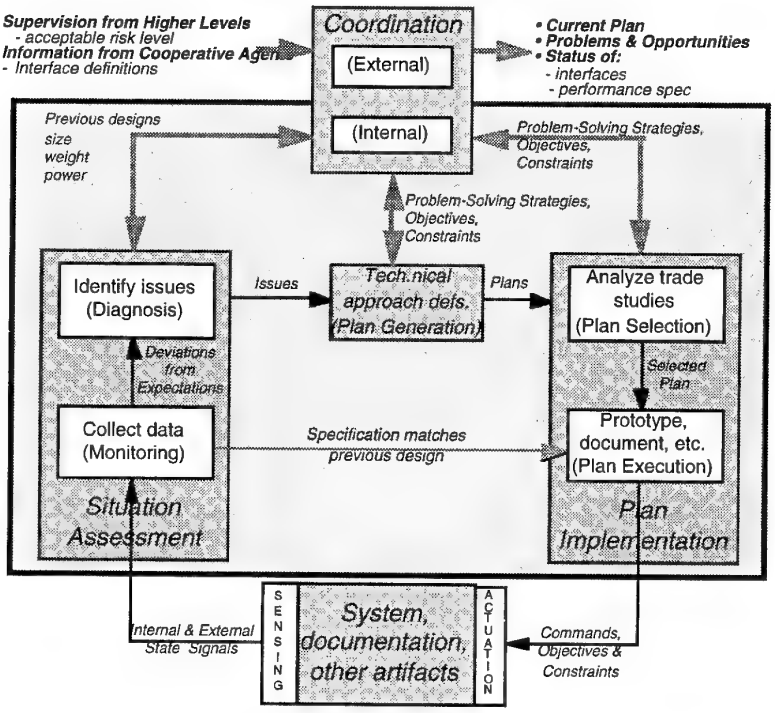


Figure 2.31 Design Functional View

For example, designers often focus on testing for their own requirements, without regard for field faults. Testing of any kind is designed late in the process since, after all, it is not the "primary" function of the product. This tendency is changing with curricula in universities and continuing education including courses entitled "Design for Testability"' or "Design for Maintainability." Aids which focus the designer's attention on this major driver of support costs have a large impact on overall avionics system costs.

2.3.3.2 Fabrication (Figure 2.32)

AI applications can produce benefits in fabrication by improving fault detection coverage and by reducing fault isolation time. The test time is not as much a concern as is the isolation time since fault isolation is frequently performed away from the automatic test equipment (ATE). Improvements in inherent testability of the IC, card, box or system all reduce fabrication costs. Without a high level of inherent testability, faults can pass undetected and result in expensive rework.

Improvements in testability will also reduce costs associated with Automatic Test Program Generation, Test Program Sets, and engineering debugging at all levels. During the fabrication phase, AI applications can support an

aggressive program of false alarm and intermittent alarm reduction by design changes and modifications to the BIT and system integrated test these reductions result from capture and analysis of faults during the intensive testing that early production run articles are subjected.

2.3.3.3 Maintenance (Figure 2.33)

2.3.3.3.1 Support Spares

Support spares are initially purchased for support and maintenance facilities and to fill the logistics pipeline. Fewer spares need to be purchased initially if it is known that improved diagnostic capability will result in fewer removals caused by false and intermittent alarms.

2.3.3.3.2 Replacement Spares

Replacement spares include cards and boxes purchased to replace the support spares that are broken or destroyed during tests as well as those spares needed to replace units with undiagnosable problems of either a hard or intermittent nature. Improving the fault isolation capability will decrease the requirement for replacement spares. Improved fault isolation also reduces the number of units wastefully destroyed during test due to "shotgun" replacement of components as an isolation technique.

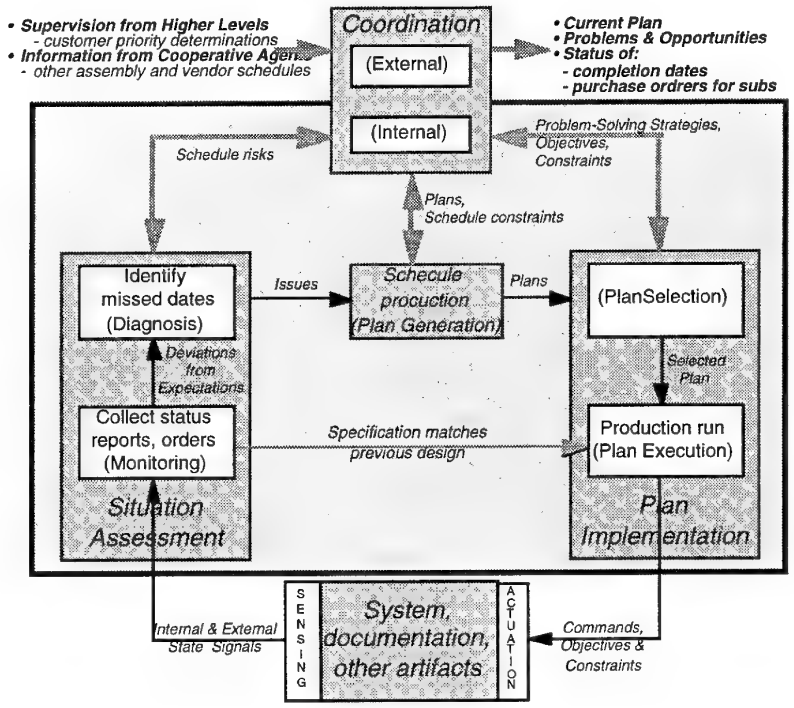


Figure 2.32 Fabrication Functional View

2.3.3.3.3 Maintenance Personnel

For avionics systems currently in the field, there are typically three groups of maintenance personnel involved in the support function. First, there are the flightline maintenance technicians who perform diagnosis to the line replaceable unit or module (LRU/M). Here we will use the term LRU for simplicity, meaning any line-replaceable element. The presumed faulty LRU is sent to the next maintenance group, the intermediate shop, which runs the LRU through ATE checks to isolate the faulty circuit card or other assembly. The isolated element is finally sent to the depot for diagnosis and isolation to the component level. Clearly, the largest factor affecting a reduction in maintenance personnel costs is the reduction in field removals. If units are not removed, they need not be tested, thus reducing both personnel requirements and required sparing levels.

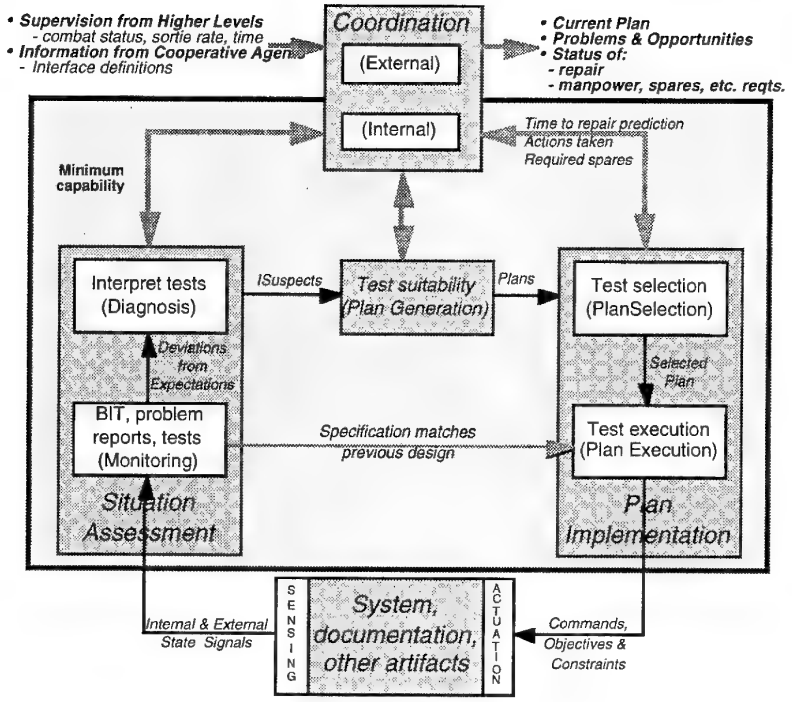


Figure 2.33 Maintenance Functional View

2.3.3.3.4 Repair Material

The factors affecting repair material are the same as those for replacement spares, except that decreasing intermittent and false alarms will not significantly decrease repair material. While a false alarm can force the replacement of a box, it usually will not cause a component to be removed from a board since causes of false alarms are usually at a higher level in the system hierarchy than the component.

2.3.3.4 Cost Drivers for Maintenance

Some maintenance will always be necessary. State of the art efforts to reduce maintenance costs focus on having equipment or system reliability (availability) sufficient to schedule necessary maintenance rather than perform it on-demand. A concept that is starting to receive attention is dispatch-with-fail (having sufficient redundancy to allow a partial avionics complement satisfy the minimum equipment list.)

A surprisingly large percentage of maintenance, however, is unnecessary. It is often driven by a flight crew "squawk" for a problem which is either not repeatable or is not a fault. The maintenance crew, however, performs some action to assure the flight crew that the "problem" has been fixed. This unnecessary on-demand maintenance accounts for up to 40% of equipment removals, called *unnecessary removals* (URs). The problem is so pronounced that some airframers maintain statistics on the *mean time between unnecessary removals* or MTBUR.

2.3.3.5 Unnecessary Removals (UR) Causes

The primary causes of unnecessary removals are thought to be due to:

- Inability to reproduce the failure environment accurately
- Presence of intermittent failures
- Inability of built in test (BIT) to correctly detect and isolate a failure

A study performed at twelve U.S. Air Force bases uncovered several other contributing factors:

- Ineffective BIT

- Ineffective technical orders
- Test equipment differences (depot vs. intermediate ATE)
- Ineffective or missing test equipment
- Inadequate skill
- Ineffective supervision/support
- Management directives
- Inadequate feedback between flightline, intermediate and depot levels
- Inaccessibility of LRUs

The potential of KBS to address these factors is discussed in Section 2.3.6.

2.3.4 Life Cycle Costs of KBS Systems

A system or tool which uses knowledge-based or artificial intelligence techniques has a life cycle cost picture which is different in some ways than the traditional software engineering development picture. Chapter 5 contains a more complete examination of tools available to assist this development, and Section 6.1.2 discusses an example of such a development. Figure 2.34 shows one view of this development cycle. There is often a higher front end cost due to prototyping. The maintenance costs may also be higher than expected due to the effort involved in keeping the knowledge current.

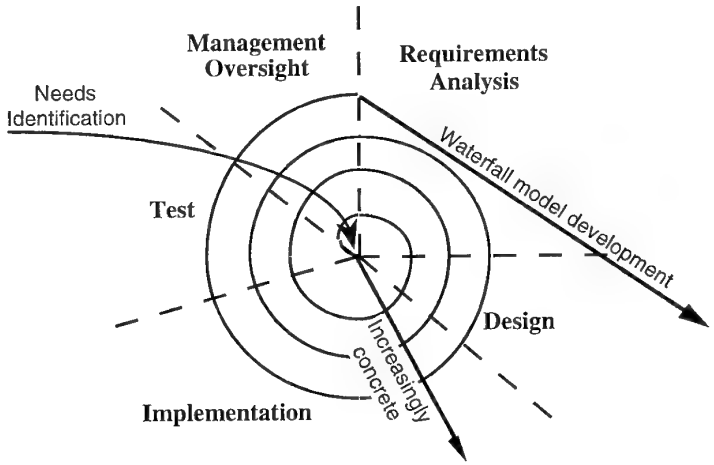


Figure 2.34 Knowledge Engineering Life Cycle

2.3.4.1 Prototyping

Systems that use artificial intelligence techniques often address less well understood problems than those using traditional, procedural or algorithmic approaches. If the problem solution *were* well understood, the solution approach would probably follow the standard software engineering waterfall model of: requirements specification, top level design, detailed design, implementation and test. For less well understood problems, however, it is often beneficial to consider various solution approaches on small scale problem cases in order to test empirically how well they address the problem. Section 6.3.2 presents a case study of this sort of approach. When viewed by someone accustomed to traditional software engineering, this rapid prototyping approach is sometimes seen as unproductive toward the goal of producing the final product. When used properly, the rapid prototyping approach is often the *only* way to proceed effectively toward the development of a solution. Improper use of rapid prototyping, such as the common practice of **using the last prototype as a software baseline**, can result in substantially higher maintenance costs due to code that was never intended by the authors to be fielded in a production system.

2.3.4.2 Knowledge Maintenance and Translation Complexity

There might be an expectation, perhaps because knowledge based software is state of the art or because it behaves intelligently, that the software will not require much support and maintenance after it is complete. The reverse is more often the case. For example, the maintenance team for the R1/XCON project at DEC is the same size as the original development team. In other words, it can take as much effort to keep the knowledge used by that

expert system current as it did to capture the information in the first place. Nonetheless, knowledge-based systems continue to be used and maintained in a variety of application areas because their benefit outweighs the cost of maintaining the knowledge base.

The initial capture and subsequent maintenance of knowledge used by an expert system is the most costly aspect of the knowledge based system's life cycle. In early knowledge-based system developments, the knowledge engineer often became an expert in the domain, and the expert became an expert in knowledge based systems. While this can be personally rewarding for both parties involved, it is *not* cost effective. On a more recent effort to develop the Flight Control Maintenance Diagnostic System (FCMDS) for the F-16, over 60% of contract funds were spent performing knowledge capture.

There are three common approaches to reducing this cost:

- Use existing on-line knowledge or data and translate it into the desired form
- Automatically "create" knowledge by machine learning techniques
- Use knowledge capture aids to extract the desired knowledge from domain experts

For knowledge capture aids, the effort and expense of performing knowledge capture are proportional to the difference between the way that the knowledge based system represents the information and the way that the cognizant expert represents it. If the representation difference is great, a large amount of effort is required both by the knowledge engineer in performing the translation and by the expert in correcting misconceptions and errors in translation made by the knowledge engineer.

Consequently, a major goal in producing a knowledge based system is to allow the expert (knowledge maintainer) to express the information in a "native tongue", i.e., in a form that is easily understood by the maintainer. In FCMDS, this was done by:

- Using readily available design information to perform diagnosis
- Capturing design information as CAD representations

The translation effort from CAD representation to knowledge-based system representation (a modified frame language) was automated, removing the human from the loop. This task simplification both reduced the man-hour effort involved and reduced human error. As a result, the knowledge maintenance task for this effort has been approximately 1 person/year as opposed to the initial development team effort of almost 4 engineers/year. In addition, a majority of this can be accomplished by junior staff since the entry task has been simplified.

2.3.5 Conclusion

As in any management endeavor, the earlier an intervention is made to address a potential problem area, the more effective the intervention is likely to be. This has long been recognized and taught in many engineering disciplines, most recently in software engineering. The focus of the intervention or management action, though, must be on the most significant costs which will accrue for the system, which are primarily in the support or maintenance phase. Efforts to apply this principle can be seen throughout the industry: design for testability, design for maintainability, lessons-learned databases, quality management teams, and so on. For newly introduced systems, these approaches have the potential to impact life cycle costs only if the changes required in other parts of the organization are allowed to occur (for example, if engineering is given a budget and schedule which allow a design for maintainability analysis to be performed). For existing systems, early intervention is no longer possible, but cost effective applications of knowledge based technologies are still possible as evidenced in the ADAM and FCMDS efforts described in Section 6.

2.3.6 Life Cycle Challenges and KBS Potential

In a sense, major challenges in any functional area are also challenges to life cycle cost management, since the difficult problems encountered in designing a system tend to be the long term cost drivers. These difficult problems can upset schedules, place unique requirements on a system's hardware or software architecture and can require the introduction of unique solutions which are understood well only by a small design team, making them difficult to maintain. These challenges and the knowledge based approaches which have the potential for addressing them are discussed in other Sections of this report. Life cycle management also has a set of more global concerns which are shown in the following tables, grouped by life cycle phase. A few issues which are common to all phases are shown separately.

Table 2.4 Issues and Approaches Common to All Life Cycle Phases.

Issue	KBS Approach	Section
Schedule upsets	Incremental scheduling	4.2
Information loss	Natural Language recognition, large scale KB techniques, hypermedia	4.1, 4.6
Database interoperability	Formal intermediate language specification	4.1

Table 2.5 Issues and Approaches for Design.

Issue	KBS Approach	Section
Requirements poorly understood	Natural Language, hypermedia, large-scale KB	4.1, 4.6
Poor BIT coverage, consistency	Model-based reasoning KB intercommunication	4.3, 6.3.3 4.1
Design information loss	Natural language recognition, KB intercommunication	4.1

Table 2.6 Issues and Approaches for Fabrication.

Issue	KBS Approach	Section
Material acquisition costs	Large-scale KB techniques	4.1
Facility idle time	Constraint satisfaction, iterative improvement, multi-agent planning	4.2
Acceptance test time	Model-based reasoning, rule-based reasoning, KB intercommunication, machine learning	4.3, 4.1, 4.4

Table 2.7 Issues and Approaches for Maintenance.

Issue	KBS Approach	Section
Ineffective BIT	Uncertainty management, Neural nets	4.5, Appendix A
Ineffective Tech Orders	Hypermedia, Adaptive interfaces	4.6
Inadequate skills	Model-based reasoning, rule-based reasoning, virtual reality for training	4.3, 4.6
Incompatible databases / information	Natural language recognition, KB intercommunication, speech understanding,	4.1, 4.6
Inaccessible LRUs	Virtual reality for maintainability	4.6

2.4 References

[2.1] Antsaklis, P.J. and Passino, K.M., (eds.), *An Introduction to Intelligent and Autonomous Control*, Kluwer Academic Publishers, Boston/Dordrecht/London, 1992.

[2.2] Sacerdoti, E., *A Structure for Plans and Behavior*, Elsevier, North-Holland, New York, 1977.

[2.3] *Combat Automation for Airborne Weapon Systems: Man/Machine Interface Trends and Technologies*, AGARD Conference Proceedings of the Joint Flight Mechanics Panel and Guidance and Control Panel Symposium, Edinburgh, Scotland, October 1992, AGARD-CP-520.

[2.4] Rouse, W.B., Geddes, N.D. and Curry, R.E., "An Architecture for Intelligent Interfaces: Outline of an Approach to supporting Operators of Complex Systems," *Human Computer Interaction*, Vol, 3, 87-122, 1988.

[2.5] Lasdon, L.S., *Optimization Theory for Large Systems*, Macmillan, New York, 1970.

[2.6] Haimes, Y,Y et al., *Hierarchical Multiobjective Analysis of Large-Scale Systems*, Hemisphere Publishing Corporation, New York, 1990.

[2.7] Mahmoud, M.S, Hassan, M.F. and Darwish, M, G., *Large-Scale Control Systems: Theories and Techniques*, Marcel Dekker, New York, 1985.

[2.8] Drouin, M., Abou-Kandil, H., and Mariton, M., *Control of Complex Systems*, Applied Information Technology Series, Plenum Press, NY, NY 1991.

[2.9] Albus, J. S., Lumia, R., and McCain, H., "Hierarchical Control of Intelligent Machines Applied to Space Station Telerobots," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 24, No. 5, September, 1988.

[2.10] Adams, M.B., and Beaton, R. M., "Planning and Planning Management for Autonomous and Semi-Autonomous Vehicles," *Proceedings of the AGARD Guidance and Control Panel 51st Symposium,* Knowledge Based System Applications for Guidance and Control, Madrid, Spain, Sept. 18-21, 1990, AGARD-CP-474.

[2.11] Onken, R., "New Developments in Aerospace Guidance and Control: Knowledge-based Pilot Assistance," *IFAC Symp. on Automatic Control in Aerospace*, München, 1992.

[2.12] Amalberti, R.; Deblon, F., "Cognitive Modeling of Fighter Aircraft Process Control: A Step towards an Intelligent Onboard Assistance System," *Intern. Journ. Man-Machine Studies*, 1992.

[2.13] Rasmussen, J., "Skills, Rules and Knowledge; Signals, Signs and Symbols, and other Distinctions in Human Performance Models," *IEEE-SMC-13,* No. 3, 1983.

[2.14] *New Advances in Mission Planning and Rehearsal System,* AGARD Lecture Series, October 1993, AGARD-LS-192

[2.15] AGARD Advisory Report "Mission Planning Systems for Tactical Aircraft (Pre-Flight and In-Flight)". AGARD AR-296 1991.

[2.16] "An Overview of the Advanced Automation Program," Department of Transportation, Federal Aviation Administration, Document Number FAA-DOT/FAA/AAP-85-2, National Technical Information Service, Springfield, VA 22150, March 1985.

[2.17] "EATMS - European Air Traffic Management System," EUROCONTROL - European Organization for the Safety of Air Navigation, Issue 1.1, Bruxelles, 16 June 1992.

[2.18] "CATMAC - Operational Concept for the Provision of Air Traffic Management Services in the Federal Republic of Germany," DFS - Deutsche Flugsicherung GmbH, Frankfurt am Main, May 1990.

[2.19] Darroy, M., "Influence of a New Generation of Operations Support Systems on Current Spacecraft Operations Philosophy: the Users Feedback," *Proceedings of SPACEOPS 92*, Second International Symposium on Ground Data Systems for Space Mission Operations, Pasadena, California, November 16-20, 1992.

[2.20] Lecaout, , "Advanced Support Tools for Space Operations and Logistics, the OPSWARE concept," *Proceedings of 4th Workshop on Artificial Intelligence and Knowledge-Based Systems for Space*, ESTEC, Noordwijk, the Netherlands, 17-19 May, 1993.

[2.21] Lee, R. E., "Logistical Impacts Within the Cost Analysis Community" - Armed Forces Comptroller, Vol. 28, No. 2, Spring 1983

[2.22] Rue, H. D. and Lorenz, R. O., "Study of the Causes of Unnecessary Removals of Avionic Equipment" RADC-TR-83-2, Rome Air Development Center, Griffiss AFB, NY, January 1983.

CHAPTER 3

HUMAN FACTORS AND FUNCTION ALLOCATION TO HUMAN AND MACHINE

3.1 Introduction

Guidance and control systems in military aircraft are essentially tools designed to support a variety of human activities in pursuit of particular military objectives. One persistent trend in the field of guidance and control of aircraft has been an increasing degree of automation of many functions previously performed by unaided humans or larger teams of humans (crews). This trend, driven by both the dynamics and complexity of modern air combat, has resulted in widespread implementation and acceptance of automated system functionality in military aircraft.

The operational importance of automation in these systems has been so evident that it has supported nearly uncritical attempts to automate virtually any function to which existing technology could lay claim. Thus, successful guidance and control automation can be seen in forms ranging from relatively subtle and straightforward improvements in flight control systems (stability augmentation) to systems that provide direct automatic control of the aircraft (auto pilots).

However, recent attempts to apply similar technology to automate sophisticated higher order functions have not produced the desired improvement in the overall system and have frequently introduced undesired side effects. For example, early attempts to automate tactical information functions in F-16's and F-18's were surprisingly unsuccessful. In this case, menu driven controls and displays, tended to segment and separate related information and resulted in a cockpit environment too difficult to manage. The effort required to obtain even important information from the system was so great and confusing that many experienced pilots elected to "turn everything off" and fly without any advanced tactical aiding.

In part, these unsatisfactory results may be attributed to an incomplete description of the domain. To some degree, this stems from the pursuit of a development methodology whose aim is to abstract a simple, well defined algorithm as a solution to the given problem. Insofar as the problem is truly simple, the technique is effective. However, when applied to problems that are not simple, and whose purpose is to cope with, and perhaps embrace complexity in the environment, such a technique is inappropriate. Many higher order functions that are current targets for automation in military aviation are, in fact, not simple; they are inherently difficult and complex.

In its current state, the art of functional allocation is heavily influenced by the perception of technical feasibility. That same perception inevitably influences the preceding functional analysis, so that "functions" themselves are typically cast in terms of technical capabilities. Moreover, even when these complex functions are properly analyzed and described, there appears to be no clear division between tasks that ought to be automated and those that should be performed exclusively by humans. Frequently the answer to the question of what to automate will depend on individual operator skill, preference or on the context of the operational situation, Thus, a strict *a priori* division of responsibility becomes a difficult, complicated problem.

Artificial Intelligence (AI) draws attention to the importance of examining the allocation of function between operator, software and hardware during the preliminary systems analysis phase [3.1]. At the 1992 AGARD Guidance and Control Panel Symposium, Taylor and Selcon pointed to the immature and imprecise nature of much of the human performance data necessary to produce such a trade off of the relative merits of man and machine components in any required cost/benefit analysis [3.2]. The approach understandably adapted by many engineering designers is to incorporate as many affordable functions as possible in a system and rely on the flexibility of the operator to integrate the remaining functions

The first major attempt to assign roles between operators and systems was that undertaken for a large-scale military ATC study undertaken by Paul Fitts following the experiences of the Berlin airlift [3.3]. Derivatives of this form of assessment of the relative merits between man and machine continued to be utilized by human factors specialists until the increasing use of computer-based automated systems necessitated the formulation of models which related man-machine performance directly to levels of autonomy of the system [3.4, 3.5]. The debate then

centered around the role of the man-in-the-loop when designers could not specify a complete automated system [3.6]. This required a further shift in emphasis to the more cognitive aspects of the man-machine interactions due to the unanticipated effect of automation increasing workload and error rate [3.7].

The advent of powerful and affordable microprocessors made possible the exploitation of the AI research emerging from the academic community, so adding a new dimension to the man-machine interface. This in turn has led to the formulation of the concept of shared intelligence linking the findings of Wiener's cybernetics, Piaget's notions of intelligence with AI, human factors and cognitive psychology [3.8]. Before such symbioses can be achieved, it will still be necessary to define the roles and responsibilities between man and machine before knowledge can be effectively shared.

This chapter attempts to examine and illuminate the problem of functional allocation for knowledge based systems. A rationale for the discussion focus on managerial functions and Decision Support Systems as the primary example of relevant knowledge based systems is presented. Human capabilities and limits and consequent implications for the design of Decision Support Systems are then discussed. Finally some general principles underlying the practice of functional allocation to knowledge-based Decision Support Systems and a philosophy for the application of that philosophy are offered.

3.1.1 Some significant characteristics of the domain:

A generalization of the military aviation domain is a broad perspective. Even when limited to operational missions, it encompasses aircraft and missions ranging from single pilot, high performance aircraft in fighter missions to large transport aircraft and crews in airlift missions to relatively low performance aircraft and special operations missions. Yet, there are some common threads that run through all these missions and influence their integral functions. These characteristics define the domain of military aviation and distinguish it, in varying degrees, from other domains in the military and in general aviation.

The military domain is intricate and complex. Even routine airlift operations must contend with variables in destination, cargo, support equipment and account for political and military factors that equivalent non-military transport operations do not address. Fighter, bomber, reconnaissance, and special operations missions are considerably more complicated.

The domain is dynamic. High speeds - absolute and relative - and the three dimensional arena combine to produce routine, rapid, and dramatic alterations of the situation. This constant change and, expectations of change, force and give emphasis to a tightly-coupled, short planning and execution cycle throughout the mission.

The domain is essentially and fundamentally competitive. There are opponents whose objective at every level of endeavor is to confound and defeat friendly plans and goals. The environment is not neutral; it is "actively" hostile. This opposition introduces a notion of deception or "intended" unpredictability that must be accounted for in functional assessment. It highlights the importance of correct judgment and the cost of errors.

Finally the domain is lethal. In many functions, the cost of even small failure is extreme. This argues for great caution and, interestingly, forces a difficult tension in an area where quick decisions and aggressive (seemingly risky) actions frequently carry the day, and hence, are the "safest" acts. An additional effect of this heightened danger it to distort "objective" evaluation and action in favor of survival of the individual at the expense of apparently higher mission objectives.

High level functions: In general, automated support in this domain typically takes form in the functional areas of (see section 2.2.1):

Assessment	(recognition and evaluation - with respect to goals - of the state of the world)
Planning	(reasoning about actions that might change the state of the world)
Execution	(acting to change the state of the world)
Verification	(post hoc comparison of the actual state to the desired state).

Guidance and control functions in aerospace systems may also be conveniently divided into two groups [3.9]. "Control" functions are directly associated with execution (flying and navigating the air vehicle). The second group "managerial" functions are related to assessment, planning, higher level execution of the plan, and verification.

Substantial advances that have been made in automating the control functions. Theoretically "unstable" aircraft, like the F-16, are routinely designed and successfully fielded as a direct result of these advances. Short of completely removing the human from the control loop, it appears that only marginal improvements can be expected in further automation of control functions.

This suggests that the greatest leverage, and hence focus of engineering development, in automation will be seen principally in addressing the managerial aspect of guidance and control functions. Supporting evidence for this view may be seen in requirements documents for most advanced military aircraft. It was also recognized in a principal recommendation of a preceding AGARD Advisory Report [3.9]. It is believed that this condition persists and thus, the principal focus of contemporary and future automation efforts is properly centered in these managerial functional areas.

Managerial Functions: A distinguishing characteristic of the managerial functions is steeply increasing task complexity. That is, the marginal growth in complexity between simple, first order tasks (calculate an aim point for weapon alpha) to the next level (hit the target with weapon alpha) is great.

Functional complexity proceeds from both the "cognitive" complexity of the task, due principally to the highly informal combination of multiple interdependent subtasks and domain complexity arising from the rapidly evolving military environment. Thus, "calculate an aim point" only requires knowledge of the kinematic capabilities of weapon alpha at some range. On the other hand, "hit the target" implies an understanding of the maneuvering capabilities of the platform, the maneuvering and destructive capabilities of weapon alpha, what effect is desired from a "hit", and the capabilities and predicted potential for the target to defeat weapon alpha in a number of ways, as well as the same appreciation for hundreds of potential targets and their latest known modifications. At the next higher level, perhaps, "plan an attack to hit the target", the task description is the proper subject of a lengthy discourse. Unfortunately, attempts to automate portions of higher order tasks, without reference to related subtasks, usually complicate rather than simplify the problem.

The integration of portions or "versions" (i.e. the non-defending target case) of a task requires a separate, additional cognitive effort that may exceed the perceived benefit or, that may cause an unacceptable delay. In part, this explains the difficulties F-16 and F-18 pilot's experienced in adapting to support systems that required them to interrupt tactical assessment and planning to manipulate menus in order to gain portions of the solutions they sought. Often it was preferable to abandon the effort to obtain improved information in favor of a timely, albeit "poorer", solution.

In spite of such experience, the increasing sophistication of users and, more importantly, demands of the environment, necessitate automation of more intricate managerial functions and hence, the development of more complex support systems. Difficulties, however, arise in determining the degree and kind of automatic support required and desired for managerial functions. Such systems are typically referred to as Decision Support Systems.

Decision Support Systems: As outlined above, complex contemporary aircraft systems require difficult decision making and problem solving activities from their users. In order to avoid limitations imposed by human cognitive capabilities efforts are under way to fully automate such tasks. This, however, is feasible only for a very limited set of well defined tasks. In most cases human cognition can not yet be emulated adequately by technical means. For the time being humans are therefore indispensable in these tasks. There is also increasing concern with respect to the reliability of computers in such systems [3.10, 3.11].

While humans currently can not be replaced by automation in certain functions, there are means to support human operators by machine intelligence decision support systems (DSS), which means *"any interactive system that is specifically designed to improve the decision making of its user by extending the user's cognitive decision making abilities"* [3.12].

For a DSS to be useful, users must be able to integrate the computer aid into their own cognitive processes. The success of, for example, an expert system depends, as Madni mentions, not only on the quality and completeness of the knowledge elicited from experts, but also on the compatibility of the recommendations and decisions with the end-user's conceptualization of the task [3.13]. In building a knowledge base there may also be conceptual conflicts between the knowledge engineer (KE) and the subject matter expert (SME). Mismatches between the developer's and user's conceptualization of the domain occur, especially with respect to individual and inter-individual differences among users. A prerequisite for DSS design therefore is the identification of what is limiting a person's decision making performance. A purely technology driven development of new automation capabilities can produce unintended and unforeseen negative consequences and should be avoided [3.14].

3.2 Human Decision Making And Problem Solving

In Chapter 2 the functions involved in guidance and control activities were identified as being monitoring, diagnosis, plan generation, plan selection, and plan execution (see figure 2.3). Looking from the viewpoint of an operator, each of these tasks requires substantial cognitive effort which usually is described by the terms "decision making" and "problem solving". This section addresses the characteristic cognitive requirements associated with both activities.

Decision making may be defined as the choice of a particular action in a situation where various actions are possible. A typical decision situation is described by the matrix in figure 3.1. The parameters in this matrix are: states of the world (e_n), available options (a_m), and consequences ($k_{m,n}$). Whenever a particular decision alternative is selected, given a state of the world, the associated consequence will follow. Each decision consequence $k_{i,j}$ may be associated with a value $w_{i,j}$ for that outcome. Utility assessments based on these values are used for selecting among competing options.

	e_1	e_2	.....	e_n
a_1	$k_{1,1}$	$k_{1,2}$	.....	$k_{1,n}$
a_2	$k_{2,1}$	$k_{2,2}$	.....	$k_{2,n}$
.	. .	.		
.	. .	.		
a_m	$k_{m,1}$	$k_{m,2}$	.....	$k_{m,n}$
	$p(e_1)$	$p(e_2)$		$p(e_n)$

Figure 3.1 Decision Matrix. (Whenever a particular decision alternative (a_i) is selected, given a state of the world (e_j), the consequence will be ($k_{i,j}$).)

Types of decision making: Some of the difficulties in making decisions become obvious when regarding the decision matrix in some more detail. Firstly the completeness of the matrix determines the degree of definition of a decision situation, i.e., whether the task is structured or unstructured. Obviously it is difficult to come up with a rational choice if not all environmental influences, options for action, and their consequences are known. Secondly the information describing the state of the world can be considered.

Depending on the probability of these data, a distinction can be made between safe ($p_{ej} = 1$), risky ($p_{ej} < 1$), and uncertain ($p_{ej} = ?$) decisions.

A third dimension of decision making refers to the available time frame. In off-line operations like , e.g., in management information systems, time usually is not a particularly critical factor. During on-line operations like, e.g., vehicle control or target recognition tasks, spontaneous reactions are requested and only seconds are available for decision making. For diagnosis and planning activities usually intermediate intervals, at least in the range of minutes, are acceptable. Figure 3.2 depicts these orthogonal dimensions of decision making with respect to time frame, environmental conditions, and degree of definition.

Decision making strategies: Mathematically optimal, i.e., rational decision making requires the selection of the decision alternative, which maximizes the utility of the consequences, i.e., if

$$EU(a_i) > EU(a_k) \quad \text{fi} \quad a_i > a_k .$$

In case of risky decisions, where only the probability of environmental conditions is known, the calculation of the expected utility (EU) takes the following form:

$$EU\ (a_i) = \sum_{j=1}^{n} w_{i,j} * p\ (e_j)$$

Deciding according to this maximal utility strategy guarantees optimality only in a statistical sense, and not for each individual decision. Humans often deviate from this maximum utility strategy and apply a two step procedure. They first tend to satisfy a subset of the most essential criteria (satisfying strategy), and only subsequently try to maximize the utility of the remaining options.

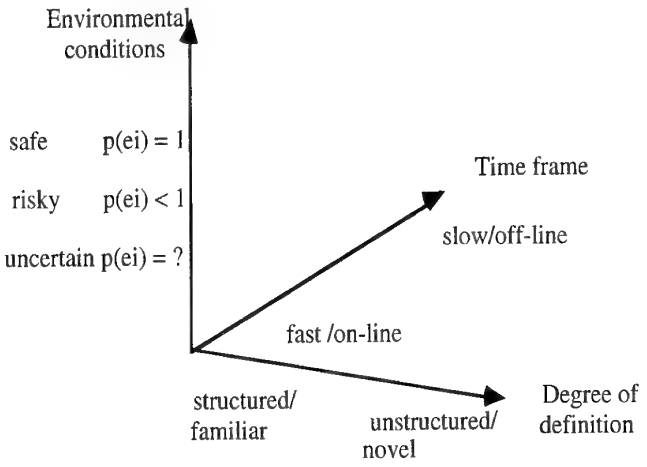


Figure 3.2 Essential Dimensions Of Decision Making

Problem solving is different from decision making in that it involves a sequence of decisions. Starting from an actual situation a sequence of actions (or thoughts) has to be pursued in order to reach a desired goal state (imagine, e.g., playing chess or performing a diagnosis on a car engine). The single state transitions involved in problem solving may be visualized as a problem space [3.15]. Figure 3.3 shows a schematic representation.

The problem space is a graph that shows system states (nodes) and transition rules (arrows) among them. This of course must include the start and goal state. Each transition within the problem space may take the form of an elementary decision as described above (figure 3.1). Since the consequences of subsequent decisions influence each other, problem solving may also be interpreted as a type of dynamic decision making.

From the problem space representation, it becomes obvious that problem solving requires two distinct activities: problem structuring, i.e. generation of the problem space graph, and searching the problem space for a success path towards the goal state (figure 3.4).

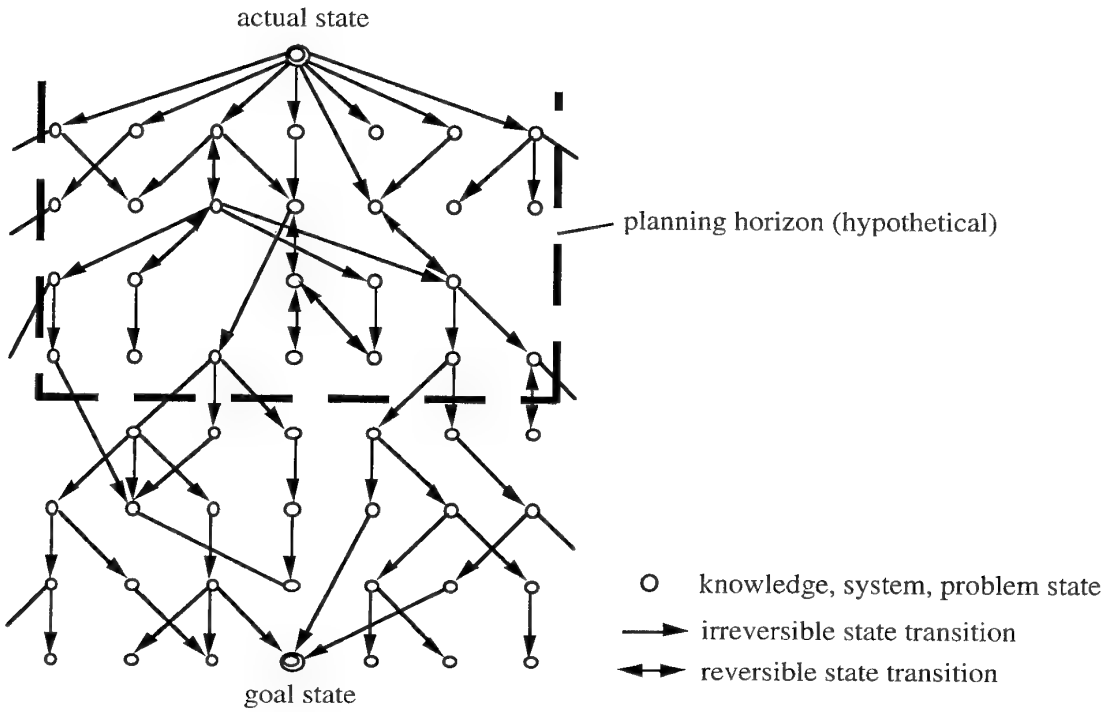


Figure 3.3. Schematic Representation of a Problem Space

Both of these activities are formidable tasks for human beings, due to narrow limitations in memory and restricted abilities in mental imagination. The mental problem of space representation, for example, can not be more accurate than the short term memory permits and mental search can not reach further than the consciously accessible planning horizon.

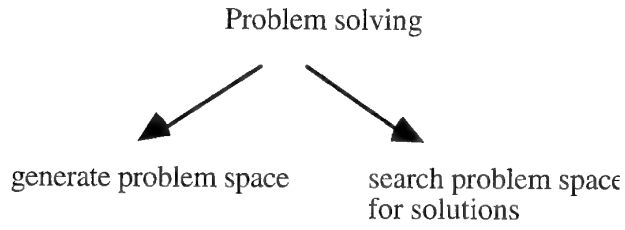


Figure 3.4. Two Aspects of Problem Solving

3.3 Characteristics and Constraints of Human Information Processing

Following the description of rational decision making and problem solving given above, an ideal, ***normatively*** acting operator should be able to:

- collect and assess information in an exhaustive and unbiased manner,
- establish a correct mental representation of a decision situation,
- calculate probabilities and likelyhoods associated with decision consequences,

- quantify and scale outcome utilities consistently,
- decide rationally, i.e., select the option with the highest expected utility,
- establish a mental model of the problem space that includes the goal state,
- find a path from actual to goal state in that problem space by mental search.

Obviously such demands are not in agreement with human nature. Operators have to operate within a framework of constraints with respect to time, knowledge, data, memory, and cognitive resources [3.16, 3.17, 3.18]. Hence real decision makers

- act impulsively,
- use a small set of rules of thumb (heuristics) as , e.g.,
  - means end analysis (pursuit of partial goals),
  - representativeness (similar situations are judged to have identical outcomes)
  - availability (similarity is determined with respect to instances available in memory)
- apply satificing criteria instead of utility maximizing criteria,
- neglect statistics and probabilities,
- have difficulty in combining competing attributes or objectives,
- use inconsistent preferences and risk assessment,
- don't consider all consequences of outcome options,
- are overly subject to situational context,
- apply bias and noise to heuristic judgment,
- have problems in analyzing or reasoning,
- have inappropriate confidence in their own decisions,
- use their internal (mental) representations whether right or wrong,
- are unable to predict processes correctly.

Humans in general use global, not analytical assessment. They make a quick assessment of a decision situation pattern, followed by an immediate categorization of the situation and an associative decision. However, in spite of the inter-individual variability in humans, a small set of recurring problems in decision making and problem solving can be identified. The most relevant of these are discussed below:

Goal-orientation: A decision maker unconsciously begins the decision making or problem solving process with the consideration of the desired output of his actions. This pursuit of a goal guides information collection and hypothesis formation. It leads to expectancy, biases and cognitive tunnel vision. In addition, the meaning of information is different for each person because of different training, experience and backgrounds. What a person perceives depends, for example, on what a person already knows.

Mental representations: Decisions are not made using the actually available data, instead data are transformed until they fit into the available mental model. Unfortunately mental models are mostly inaccurate, usually being less accurate than perceived by most people. They are applied independently of the state of training or expertise an operator might have. Consequently projections in time and space based on such mental imagery ("mind's eye") are often wrong.

Memory limitations: Reasoning can only be performed with the content of the working memory, which is no more than 5 to 9 chunks (Chunks are groups of information which are subjectively perceived as units). Coding is a suitable means to put as much information as possible into one chunk. If more information is required than fits into the working memory, it must be retrieved from long term memory. Also the speed of cognitive operations is limited (A mental comparison takes, e.g., typically 0.1 seconds).

Mental arithmetic: The brain is not well suited to perform complicated numerical operations. Hence it is almost impossible to process conditioned probabilities and utilities or to estimate risks reliably in any given situation.

Mental estimation of uncertainty does, e.g., not follow the Bayes-theorem. Instead, global estimates are used, which usually have a heavy bias.

Cognitive levels of control of human action: Depending on the task difficulty, the given time frame, and the degree of training operators show three distinctly different levels of action, namely *skill based*, *rule based*, or *knowledge based behavior* [3.19]. In this classification skill based behavior describes those actions, which are performed in a subconscious, quasi-automated manner in fractions of a second. Examples are walking, playing an instrument, or ten finger type writing, i.e., activities which require instantaneous reactions. A prerequisite for skill based behavior is long term training. Rule based behavior can be observed in cases where the solution to a problem is known, but the single steps toward that a solution must be performed consciously according to given rules. This requires no particular mental effort and takes typically 3 to 5 seconds per action (see figure 3.5). Knowledge based behavior, finally, is applied in situations, where no known solution to a problem exists. Consequently creative or innovative behavior is needed, which may take an unpredictable amount of time, while a success can not be guaranteed.

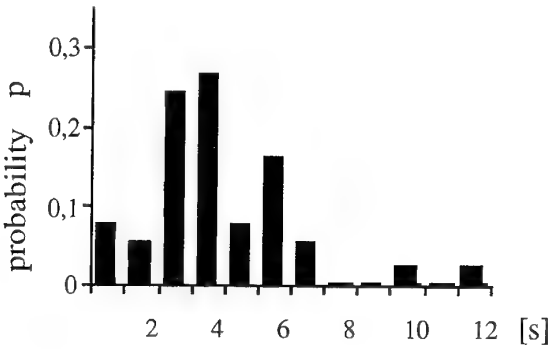


Figure 3.5. Interaction Intervals Observed During the Rule Based Use of a Dialog System [3.21].

It should be emphasized, that a particular task can be performed by a particular operator on varying levels of cognitive control, depending on his actual state of training. Also a skill achieved in a task by extensive training may be lost again if training ceases. This may lead to a fallback from skilled to rule based or even knowledge based behavior, associated with longer operating times. Furthermore it is obvious, that tasks requiring very fast reactions can only be mastered on the skill based level.

Human reliability: Humans are notorious for being unreliable. At a first glance the reasons for human error appear to be unpredictable. This however, is not true, as Norman discovered [3.20]. He distinguishes between two types of action errors with respect to their reason, i.e., *mistakes* and *action slips*. Mistakes are made due to lack of knowledge, stress, fatigue, cognitive fixation, or wrong mental representation. In contrast to mistakes, action slips are errors made in carrying out an otherwise correct intention.

Slips occur if the inappropriate schema is activated or triggered, where schema means a sequence of linked behaviors. Several types of slips may be distinguished. *Mode errors* may occur if a situation is misclassified, as when an autopilot is in a different mode of operation than expected. *Capture errors* are likely to occur if a similar, more frequent or better learned sequence captures control. *Description errors* may happen if the triggering information for a schema is ambiguous or undetected. This results in a correct operation being applied on the wrong item. From this description it is essential to note, that slips occur during periods of rule based behavior and also show the timing characteristics associated with this type of operation (compare figure 3.5).

Technical support of human cognition

As previously stated, the identification of what is limiting a person's decision making performance is a prerequisite for DSS design. Consequently a summary of design rules for the compensation of cognitive constraints is listed below, which have been derived from the described human information processing deficiencies [3.22, 3.23]:

- Improve information display:
  - make use of human Gestalt and figure perception capabilities
  - use task and situation adaptive information filtering
  - use high fidelity computer graphics and animation
  - use multi media presentation
- Improve the human-machine dialog:
  - Increase bandwidth of data exchange by novel interaction techniques (e.g. , direct manipulation and speech),
  - match dialog form to user training and cognitive state,
  - provide user guidance to reduce available options,
- Support information management:
  - reduce and facilitate routine data handling work,
  - provide bookkeeping functions,
  - reduce memory load by visualization,
- Support decision making:
  - provide decision situation structuring aids,
  - perform statistics, probability and utility calculations on computers,
- Support problem solving and planning activities:
  - provide problem space visualization,
  - provide automated reasoning functions,
  - enable man-machine goal sharing,
- Reduce educational and training requirements:
  - provide predictor information for test and exploration,
  - provide embedded training,
  - provide on-line help functions,
- Build error tolerance and error insensitivity into the system:
  - use immediate error feedback (predictors),
  - use reversible functions (UNDO),
  - use error tolerant functions (DWIM),
  - make system state always unambiguously clear,
  - use dissimilar command sequences for different actions,
  - assign difficult actions if action consequences are severe.

3.4 Design and Implementation of Decision Support Systems

Decision support systems usually are configured as depicted in figure 3.6: An operator has direct access to the system to be controlled and, in addition, may exchange information with a decision support system via the user interface. As may also be seen in this figure, the DSS itself has access to the same information about the controlled vehicle or process as the user.

Levels of automation in man machine systems:

As Seifert & Neujahr point out, the spectrum of interaction between the human and the computer reaches from full manual operation to full automation. They identify six automation levels for man-machine interface functions, which are widely accepted in the aircraft industry, and which vary with respect to the percentage of authority assigned to the human (table 3.1) [3.24]:

The full *manual* and the full *automatic* modes of operation need little explanation. Some essential human factors arguments must be mentioned however: Manual operation often leads to operator stress, high workload, and fatigue. On the other hand too much automation is likely to lead to boredom, complacency, and erosion of competence. In

critical situations, it may be dangerous to move the operator too far out of the control loop by supplying too much automation.

Some additional remarks have to be made with regard to the intermediate modes of automation, which are candidates for decision support. In the *manual augmented mode,* manual control functions are augmented by automatic control systems as, for example, nose wheel steering, and stabilizers in aircraft. Mental decisions in this mode may be supported by electronic checklists, spreadsheets, integrated displays, or reversible functions for error correction. In all these cases, the authority remains entirely with the operator.

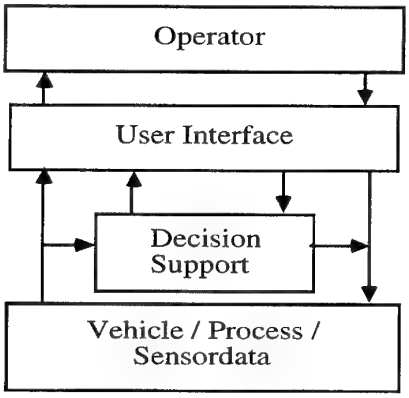


Figure 3.6. Typical Decision Support System Configuration

Examples for the *manual augmented-automatically limited mode* are data entry formatting and validation checks as well as autonomous input error correction (do what I mean, DWIM). The corresponding approach in the area of manual control is the limiting of control inputs in order to ensure system safety (Envelopes, safeguarding functioning points). Examples are angle of attack or g-level control in aircraft and traction control systems in cars. As depicted in table 3.1, the authority is partly and definitely transferred to the computer in this mode. The actual size of this share, however, need not be fixed, but can be adapted to varying situations, system states and user characteristics.

In the *automatic-manually limited mode* manual override and takeover of automatic functions is made possible. This is, e.g., the case with heading and course being controlled by autopilots in an aircraft. These systems are generally configured in such a way that the degree of human authority is at the discretion of the user and may reach from full automatic to full manual as depicted in table 3.1.

Table 3.1: Automation Levels in Man-Machine Systems:

mode of operation	authority
manual	human
manual augmented	
manual augmented-automatically limited	
automatic-manually limited	
automatic-manual sanction	
automatic	computer

The *automatic-manual sanction mode* is characterized by automatic functions with manual accept/reject capabilities. The assessment of options presented by an expert system, the approval of automatic target identification or prioritization, and the acknowledgment of a trimming computers proposal in a submarine are typical examples. The degree of human authority depends on whether manual accept/reject is optional or mandatory, and hence may cover a wide spectrum. Sometimes manual sanction, while technically possible, is not practical due to time restrictions, as is often the case in target identification tasks.

3.4.1 Principles Of Functional Allocation To Knowledge-Based Systems

For conventional systems, including conventional expert systems, the basic principles of functional allocation are quite straightforward:

- Consider the aims of the proposed automation carefully. Automation should not be introduced for its own sake.
- Allocate tasks to human or machine according to what each does best.
- Pay attention to the interaction and integration of man and machine.
- In considering aspects of a task for machine execution, do not be constrained by the pattern into which a human operator breaks the task down or by reasoning processes that a human operator adopts (or thinks he adopts).
- Involve users in the design and functional allocation from the outset. A cornerstone in the task of building an effective MMI is the involvement of the user from the outset. Besides being a useful way of gaining user acceptance, the early involvement of users in system design benefits from ideas that domain experts may offer about the sort and degree of assistance that they require.

While it clearly makes sense to apportion to the man and the system respectively those aspects of the task that each does best, there are no infallible rules to define these proficiencies [3.25]. Many papers addressing the question of aircrew workload and its use as a metric for functional allocation have been written over the past two or three decades and very often the technologies proposed for analyzing and understanding workload have centered on some form of task decomposition and analysis. Task breakdown is then, typically, the first step in considering functional allocation in combined manned and automated systems and the experience gained from workload studies based on Time Line Analysis and modern derivatives of that approach will be of great value in considering the functional decomposition of the mission.

Complications arise almost immediately and parallel those which are experienced in applying similar decomposition analysis to the assessment to workload. In that case the simplest approach is to measure the time required to perform each sub-task on its own and to derive the total time required for the whole task by adding those for the components. This approach runs into difficulties because the sub-tasks cannot be considered in isolation; even when the activity is reasonably self contained (e.g. changing the radio frequency), its execution is affected by the other sub-tasks on hand at that time and by external circumstances. The interactive effect may be favorable, in that the task may be performed quicker under pressure, or unfavorable so that it is done slower or not at all. The whole may not always be greater than the sum of the parts, but it is generally different.

Task interaction is a key issue in considering allocation to knowledge-based DSS. This is particularly true when a task might be accomplished in different ways (different sequencing of the sub-tasks) depending on the circumstances. Then a preliminary allocation of a subtask might be frequently reversed and a requirement is added for the system to "understand" situational context.

3.5 Cooperative Human-Computer System Concept

In order to identify suitable ways of interacting with computer based advice, let us, in an experiment of thought, assume that the decision support be provided by a human advisor. In what way would we like his advice being presented to us ? This line of thinking leads us to look in some more detail on the ways of information exchange among humans. Nickerson has compiled a comprehensive list of conversational attributes (Table 3.2) [3.26].

From this table the crucial role of shared knowledge and accepted conventions become obvious. Based on this a priori information humans are able to recognize distorted information by using context. It also permits a sender to

articulate ambiguously, erroneously, and incompletely and still be understood. Another attribute worth mentioning is the enormous bandwidth involved in interpersonal communication. The observation that peer status is required in order to ensure a serious conversation is of particular relevance in the context of DSS. Computer based advice will only be accepted if it is perceived as being of excellent quality.

Table 3.2: Some Characteristics of Human Conversation

Shared knowledge:	Situational context
	Common world knowledge
	Special knowledge
	History
Accepted conventions:	Bidirectionality
	Mixed initiative
	Apparentness of who is in control
	Rules for transfer of control
	Sense for presence
	Structure
	Characteristic time scale
	Intolerance for silence
Other attributes:	Nonverbal communication components
	Wide bandwidth
	Informal language
	Peer status of participants

Certainly some characteristics of conversation among humans would also be desirable for human computer interaction. The advice provided by a DSS should , for example, be made adaptive in the sense that it is tailored to the actual situation, to the status of the system, and to the operator behavior and capabilities (This includes in particular sensory, motor, and mental workload as well as cognitive level of control and motivation). An intelligent assistant of this kind would allow the user to use a system in a manner he wishes, but surrounded by a multidimensional warning and alerting system, an *electronic cocoon* that informs and supports him if he is approaching some limit (management by exception) [3.27, 3.8].

In order to enable a computer to behave as a human-like partner, it must obviously be provided additional information. Building up shared knowledge requires the storage of knowledge bases and procedural rules. In addition the machine must know about strategic goals and intentions. This allows the computer to check operator inputs and system outputs to determine if they are logically consistent with the overall goal. It also provides mutual monitoring of man and machine. A system that embodies an expectancy of the kinds of errors that might occur may also be able to detect errors automatically as deviations from normative procedures. Systems with such capabilities are said to be *goal sharing* and *intent driven* [3.28, 3.29]. An example of such a system is described in chapter 6.1.1.

A possible structure for such cooperative human computer systems is presented in figure 3.7. It is based on proposals made by Rouse and colleagues [3.6]. As may be seen from comparison with figure 3.6, a set of three data bases has been introduced into the system. Stored in these data bases is knowledge about the state of the world, about the system status, and about the human operator state and goals.

First, in order to identify operator state and goals, a collection of active goals, plans & scripts is needed. Second, knowledge about system functioning is encoded in specific scripts and procedures. Finally, environmental conditions and operational phases describe the state of the world. By making use of this knowledge an intelligent assistant may provide adaptive behavior at the user interface, where adaptivity may be desirable with respect to environmental conditions, operational phases, and system states. Adaptivity to user performance characteristics

requires inputs from an *operator model*. Main functions of this intelligent assistant are *interface management* and *assistance functions*.

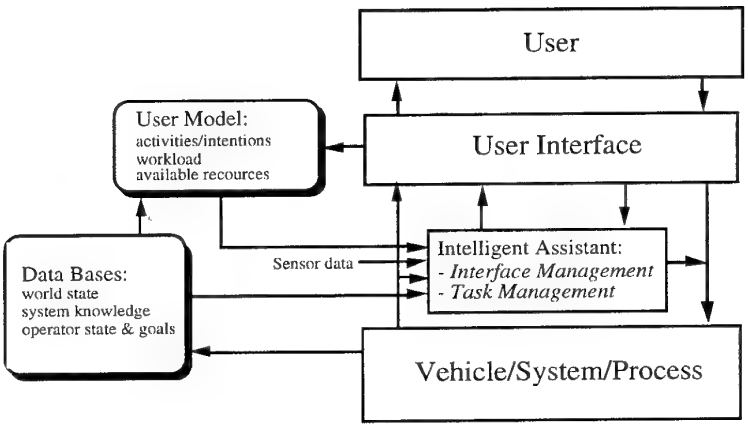


Figure 3.7. Cooperative Human-Computer System Architecture

Interface management, i.e. the way data are exchanged at the user interface, can be shaped according to the dimensions shown in Table 3.3.

With the ever increasing amount of information available in automated systems, the role of *information management*, i.e., storage, organization, retrieval, and filtering becomes more and more critical. It is no longer acceptable to confront a user with an enormous amount of raw data. Instead he must be supported by preprocessing and filtering functions. Modern computers also offer various new possibilities for *information display*, which were not available before [3.30]. Among these are color and dynamics, which allow very efficient means for the interactive evaluation of large sets of data. *Dialog design* also is a particularly essential aspect of interface management. Task scheduling, user guidance and the selection of a suitable dialog form are efficient means to match the way of data exchange to the needs of the user. One especially efficient form of a dialog is direct manipulation, which allows the user to point directly to graphical objects on a screen.

Table 3.3. Dimensions of Interface Management

information management	storage, organization, retrieval, filtering.
information display	coding, integration, selection of modality.
dialog design	scheduling, user guidance, dialog form.

Assistance functions may take the form of *decision support*, and/or *error management*. The spectrum of generally used algorithms in DSS is compiled in table 3.4. It is not the purpose of this paper to explain all these

algorithms. The reader is referred to chapters 4 and 6 of this report and to some referenced publications [3.12, 3.31, 3.32, 3.33].

A special aspect of user assistance is *error management*. The goal is to build error tolerant or error insensitive systems by the automated identification, compensation and prevention of errors. Errors of omission or commission may, e.g., be identified and compensated by making reference to stored legal procedures [3.34].

Table 3.4. Candidate Algorithms For Decision Support

Algorithm	*Supported decision making aspect*
Process modeling	Predictions for future or different conditions.
Value modeling	Combination of attributes.
Machine learning	Compensation of systematic inconsistencies or biases for judgment refining and amplification.
Adaptive aiding facilities.	Automated reasoning
Numerical techniques	Calculation of conditioned probabilities and utilities.
	Searching for success paths in problem spaces.
Symbolic techniques	Generation and validation of hypotheses by rule based forward and backward inferencing.

If the interface management and assistance functions mentioned above are to be adaptive, input is needed with respect to the system operating phase, the state of the world, and, last but not least, the operator. In particular the identification of operator characteristics is a very difficult task to be performed by an *operator model.* This refers to an operator's

- goals and intentions
- utilization of sensory, motor, and mental resources
- objective and subjectively perceived workload.

As Rouse and colleagues mention, the determination of an operator's goals and intentions can be acquired explicitly by asking the operator for input [3.6]. Implicit determination of goals and intentions is much more difficult. Obvious sources of information about the operator are - except from physiological measures - the overt observable actions performed at the interface. The identification of intentions then requires inference processes in the context of a particular task and system situation. Reference to legal procedures which are stored in the computer also helps to discriminate expected (explained) from unexplained actions (errors or innovations).

Continuous observation is a particular interesting approach to identify individual user behavior and preferences [3.35]. The basic idea is to copy user behavior by applying machine learning algorithms (figure 3.8). As each user has a different mental model and experience, the computer can accumulate expertise from several users. Consequently the interface becomes smarter during operation.

Resource utilization (auditory/visual, verbal/spatial, cognitive, or motor) may be derived from current and projected on-line workload analysis. Actual resource utilization may be used as a main determinant for interface

management. Operator models then act as an on-line expert system for the design of displays and programmable controls. In addition a prediction of performance in current and potential future tasks can be based on model outputs.

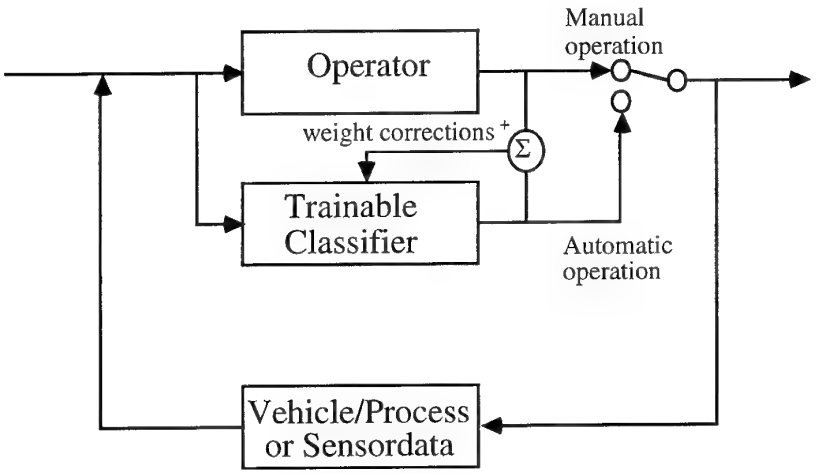


Figure 3.8 Identification Of Operator Characteristics By Observation

3.6 Operational Aspects of Decision Support Systems

As mentioned earlier, it is very essential that a DSS is matched to the cognitive processes and the mental task conceptualization of it's user. This is not a trivial task because the spectrum of operator behavior is rather broad. Operators may, e.g., be naive or expert, either in computer handling or in the subject matter, or in both. In addition their use of the DSS may be casual or frequent (figure 3.9). Thus one operator may show skilled behavior and another rule based behavior in identical tasks. The actual level of action applied to a particular task depends on the task difficulty and the level of training and expertise (see section 3.2). Also a skilled operator will, after a long break, have to resort for a while to rule based or knowledge based behavior until he is back to a reasonable level of training. As table 3.4 shows, there is also a characteristic time scale associated with different levels of action.

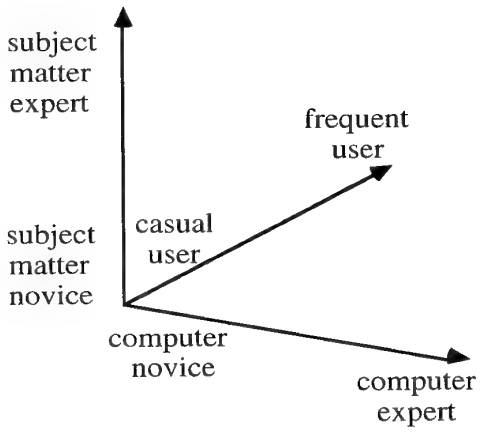


Figure 3.9 Some Relevant User Dimensions

Some tasks, like playing chess, are so demanding, that even after extensive training, they remain knowledge based tasks. On the other hand there are tasks which, due to limitations in execution time, must be performed at the

skill level. Among these are reaction, stabilization, and detection tasks. Table 3.5 lists a number of tasks ranging from guidance to communication which can not unambiguously be assigned to an action level. Diagnosis, e.g., can subsequently imply all actions levels depending on individual and inter-individual peculiarities as well as on task characteristics.

Table 3.5. Cognitive, Technological and Task Related Aspects of Decision Support Systems

Available time	Level of action	Operator task	Decision support	Properties	Implementation
t/s 100 10 1 0,1	Knowledge based Rule based Skill based	Communicat. Navigation Diagnosis Supervision Identification Classification Recognition Guidance Stabilisation Reaction Detection	Information-mgnt Book keeping Situation structuring Visualization Risk evaluation Option evaluation Teaching Consulting Automatic takeover Display of status, command or predictor information	selfexplaining transparent adaptive cooperative intuitive reactive Mode of information presentation: head-up peripheral acoustical tactile	symbolic DP sequential DP numerical DP parallel DP

The support of rule and knowledge based action is with respect to information management, decision support, and error management. The time available usually allows a dialog between the user and the DSS. The support can take several forms, corresponding to the modes of operation described in table 1.

In the "manual augmented"-mode the user can get mainly information management support, i.e., help in handling large amounts of information and in structuring a decision situation. This includes task and situation dependent filtering and display of information using advanced computer graphics features like color and animation. Other aspects of this approach is the visualization of decision and planning situations.

The "manual augmented-automatically limited"-mode involves no dialog with the user. Input limiting or takeover occur automatically without additional notification. The same applies to error identification, compensation, and prevention.

Decision support in the "automatic-manually limited"-mode involves the delegation of tasks to automation with subsequent supervision. This is achieved simply by the selection of the desired operation modes. A real dialog between user and machine does not take place.

Decision support in the sense of advice giving, consulting, or teaching takes place in the "automatic-manual sanction"-mode. The assessment of options presented by a DSS usually requires some additional inquiry about the validity, background and context assumptions implied in the computer solution. Hence it is not enough for a DSS to come up with a simple suggestion. Instead, it should be transparent, self-explaining, cooperative and adaptive to task, situations, and operational needs.

Self explanation in rule based expert systems usually takes the form of long lists of rules which basically permit one to trace back a hypothesis to the causal facts. However, this is frequently considered to be too tedious a task. Consequently, complacency and overconfidence can be observed and decision support is accepted without further check. Therefore, a desirable design goal for DSS is to give the user as complete a picture of what is going on in the

computer as possible. One obvious way to go is to replace textual information, as far as possible, by graphical displays which can be read and understood much faster. At this instance of time very few graphical DSS examples exist. One of the few is depicted in figure 3.10, showing a decision parameter display for a risky decision among three alternatives. As may be seen, the computer proposes to select alternative 2 , based on expected utility calculations. However, instead of just saying " select a_2" the whole situational context is presented. The user can follow the movement of the probability pointer, while the environmental conditions e_i change. As the pointer approaches the border where the expected utility of a different decision becomes greater than that of the actual one, he can actually see, on which information the computer suggestion is based and what the likely outcome and other options are.

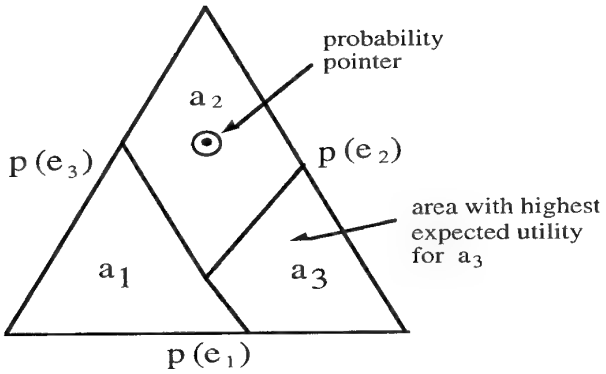


Figure 3.10 Decision Parameter Display for a Risky Decision Among Three Alternatives [3.36]

3.7 Conclusions and Recommendations

This chapter addressed cognitive human factors aspects of guidance and control with an emphasis on task allocation to human and machine and decision support.

Basic principles of function allocation to human and machine were identified (see section 3.3)

- Automation should not be introduced for its own sake.
- Allocate tasks to human or machine according to what each does best.
- Pay attention to the interaction and integration of man and machine.
- Machine execution of a task must not emulate the human reasoning processes.
- Involve users in the design and functional allocation from the outset.

From a detailed analysis of cognitive behavior the following recommendation for the technical support of human cognition were derived (see section 3.2):

- Improve information display:
- Improve the human-machine dialog:
- Support information management:
- Support decision making:
- Support problem solving and planning activities:
- Reduce educational and training requirements:
- Build error tolerance and error insensitivity into the system.

Possible implementations of Decision Support Systems (DSS) were reviewed, in addition some example solutions were discussed. It was shown that a purely technology driven development of new automation capabilities can produce unintended and unforeseen consequences. In order for a DSS to be useful as a cognitive prosthetic, users must be able to integrate the computer aid into their own cognitive processes. A prerequisite for DSS design therefore is the identification of what is limiting a person's decision making performance.

Currently there is still little information concerning the efficiency of DSS available. The accessible data show however a general tendency of increased performance. As far as the acceptance of DSS is concerned, reliability appears to be a very critical factor especially if accuracy changes without warning.

A general difficulty in interacting with DSS results from the observation that decision support functions often are not well understood by operators. There is a false belief in the objectivity of computer results. In many cases there is also an obscured responsibility distribution between man and machine.

Since the reason for a wrong DSS diagnosis mostly is not a programming error but rather wrong background assumptions, the interface layout is a very essential part in DSS design. It must provide the user with sufficient information about the functioning of the applied decision support algorithm including the underlying assumptions. Of course, this information must be made easily accessible and readable by using suitable display formats. Only such transparency and self explaining features will enable a critical evaluation of DSS suggestions, and a qualified manual takeover in case of DSS failure.

3.8 References

[3.1] NATO STANAG 3994 AI, The Application of Human Engineering to Advanced Aircrew Systems Design.

[3.2] Taylor R. M., Selcon S. J. (1992), Operator and Automation Capability Analysis: Picking the Right Team. In: Proceeding of NATO AGARD Joint FMP/GCP Symposium, Edinborough U. K.

[3.3] Fitts P. M. (1951), Human Engineering for an Effective Air Navigation and ATC System. NRCC, Washington d. c.

[3.4] Sheridan T. B. (1984) Supervisory Control of Remote Manipulators, Vehicles and Dynamic Processes, in: Advances in Man-Machine Systems Research, JAI Press

[3.5] Sheridan T. B. (1992) Telerobotics, Automation, and Human Supervisory Control. The MIT Press, Cambridge, Massachusetts.

[3.6] Rouse, W.B.; Geddes N.D. & Curry, R.E. (1987). An Architecture for Intelligent Interfaces: Outline of an Approach to Supporting Operators of Complex Systems. In: G. Salvendy (Ed.), *Human Computer Interaction, Vol.3* (pp. 87-122). London: Lawrence Erlbaum .

[3.7] Woods D. D. (1986), Cognitive Technologies: The Design of Joint Human-Machine Cognitive Systems., AI Magazine 6(4).

[3.8] Boy G. A. (1991), Intelligent Assistant Systems, Academic Press. San Diego

[3.9] AGARD (1986) Improved Guidance and Control Automation at the Man-Machine Interface, AR 228

[3.10] Bellin D.,Chapman G. (1987) Computers in Battle - Will They Work ? Harcourt Brace Jovanovich, Boston.

[3.11] Greeley, B.M. (1988) Pentagon Vincennes Inquiry Backs Commander's Actions. AW & ST, Aug. 29, 21-22.

[3.12] Zachary W. W. (1988) Decision Support Systems: Designing to Extend the Cognitive Limits. In: M. Helander (Ed.): Handbook of Human Computer Interaction, Chapter 47, pp. 997-1030, Elsevier.

[3.13] Madni A.M. (1988) The Role of Human Factors in Expert Systems Design and Acceptance. Human Factors Special Issue: Expert Systems 30,4,395-414

[3.14] Woods D.D. & E.M.Roth (1988) Cognitive Engineering: Human Problem Solving with Tools, Human Factors Special Issue: Expert Systems 30,4,395-414

[3.15] Newell, A., Simon H.A.(1972) Human Problem Solving. Prentice Hall, Englewood Cliffs, N.J.

[3.16] Hink R.L. & D.L.Woods (1987) How Humans Process Uncertain Knowledge: An Introduction for Knowledge Engineers. The AI Magazine, Fall,41-53

[3.17] Wickens C.D. & J.M.Flach (1988) Information Processing. In :(Wiener E.L. and D.C. Nagel eds.) Human Factors in Aviation. Academic Press, pp.263-301.

[3.18] Nagel, D.C. (1988) Human Error in Aviation Operations. In :(Wiener E.L. and D.C. Nagel eds.) Human Factors in Aviation. Academic Press, pp.263-301.

[3.19] Rasmussen J. (1983) Skills, rules, knowledge: signals, signs, and symbols and other distinctions in human performance models. IEEE Trans. Syst. Man Cybern. SMC-13 (3), 257-267.

[3.20] Norman D.A. (1981) Categorization of action slips. Psychological Review, 78, 1-15.

[3.21] Kraiss K.-F. (1993) Mensch-Maschine Dialog, In (Schmidtke H. (Hrsg.) Ergonomie. 3. Auflage, Verlag Hanser.

[3.22] Kraiss K.-F. (1989) Human Factors Aspects of Decision Support Systems, In: Operational Decision Aids for Exploiting or Mitigating Elektromagnetic Propagation Effects, NATO, AGARD CP 453, p 3.1 - 14.

[3.23]]Rouse W.B. (1988) Adaptive Aiding for Human/Computer Control. Human Factors Special Issue: Expert Systems 30,4,395-414

[3.24] Seifert R., Neujahr H.(1987) Moding Strategy for Cockpit Data Management in Modern Fighter Aircraft. AGARD-GCP/FMP Symposium, Stuttgart, September.

[3.25] Billings C. (1991) Human-Centered Aircraft Automation: A Concept and Guidelines, NASA TM 103885, August.

[3.26] Nickerson R.S.(1976) On conversational Interaction with Computers. In: S.Treu (Ed.) User-Oriented Design of Interactive Graphics Systems. Proc. of ACM/SIGGRAPH Workshop, Pittsburgh, Pa. ,101-113

[3.27] Speyer J.J. (1988) Towards Design-Induced Error Tolerance, In: Proceedings of the Int. Conf. on Human-Machine Interaction and Artificial Intelligence in Aeronautics and Space, Toulouse-Blagnac, 28-30 September, 69-94

[3.28] Wiener E.L. (1988) Cockpit Automation. In :(Wiener E.L. and D.C. Nagel eds.) Human Factors in Aviation. Academic Press, pp.263-301.

[3.29] Shalin V.L., Perschbacher D.L. & P.G. Jamar (1988) Intent Recognition: An Emerging Technology, In: Proceedings of the Int. Conf. on Human-Machine Interaction and Artificial Intelligence in Aeronautics and Space, Toulouse-Blagnac, 28-30 September, 125-137

[3.30] Ellis S.R. (1987) Spatial Displays and Spatial Instruments, NASA CP 10032

[3.31] AGARD (1991) Knowledge Based System Applications for Guidance and Control, CP 474

[3.32] AGARD (1992) Machine Perception, Lecture Series 185

[3.33] AGARD (1992) Advanced Aircraft Interfaces: The Machine Side of the Man-Machine Interface, CP 521

[3.34] Woods D. D. (1990), Modeling and Predicting Human Error in Human Performance, in: , (Elkind Ed.) : Models for Computer Aided Engineering, Academic Press, London.

[3.35] Freedy A., Madni A., and Samet M. (1985) Adaptive User Models: Methodology and Applications in Man-Computer Systems. In: Rouse W.B. (Ed.): *Advances in Man-Machine Systems Research, Vol. 2*, JAI Press Inc., Greenwich, Conn., 249-293.

[3.36] Amey D.M., Feuerwerger P.H. and R. Gulick (1979) Documentation of Decision Aiding Software: OPINT Users Manual. Decision and Designs, Inc., McLean, VA.

CHAPTER 4

ENABLING TECHNOLOGIES AND THEIR POTENTIAL

4.1 Introduction

The purpose of this chapter is to survey a variety of emerging computational technologies that we believe may be of significant use in building future systems for aerospace guidance and control. It should be noted that we have used the term "knowledge-based systems" quite loosely when considering suitable topics; all sub-areas of artificial intelligence and computational human factors were deemed relevant. This includes several topics, like fuzzy logic and neural network methods, that sprung from artificial intelligence research but have recently developed infrastructures (conferences, journals, professional societies) of their own.

The chapter is organized into seven major technology sections. The first deals with the heart of knowledge-based systems, the knowledge itself. More specifically, we discuss methods for acquiring, representing, and managing knowledge over the potentially many year life-cycle of a complex aerospace mission. The second section describes combinatoric manipulation of knowledge within a *search space*. We include a detailed look at one of the most important classes of problem-solving within a search space, that of planning and scheduling. The third section treats knowledge-based technology for reasoning about complex physical devices and processes. It covers first-generation, experiential or *expert systems* as well as second-generation *model-based* reasoning mechanisms that attempt to understand physical systems at the level of a trained engineer. The fourth section surveys both "traditional" formal methods for computational control of aerospace systems as well as knowledge-based or "symbolic" control methods. The fifth section discusses an extremely active area of current artificial intelligence research, that of machine learning. This section includes a brief overview of neural methods (described in more detail in Appendix A). Learning can be applied to all phases of knowledge-based systems technology; we provide a guide to some of the most important aerospace applications. The sixth section discusses the mechanisms used by knowledge based systems to gather information from the external (both natural and artificial) environment. Finally, in the last section, we describe technology for facilitating human interaction with knowledge-based systems within aerospace missions.

Technology progresses from scientific research on basic concepts to applied research on specific aerospace domain problems to final applications in aerospace missions. For much of the work discussed in this chapter, generic tools have been constructed to allow easy application of the technology to other, related missions. We have noted particular areas where tool development is important. Throughout this chapter we will illustrate knowledge-based systems technology with brief examples from actual aerospace missions; Chapter VI of this report will provide a more detailed look at several integrated aerospace applications which employ this technology.

For all of the topics discussed below, it is also important to keep in mind three overall and interacting constraints on their practical applicability in aerospace domains. First is the issue of connectivity with other sources of knowledge like sensor data streams and existing large-scale databases. All of the systems discussed in this report need to operate effectively in an information-rich environment. Second is the problem of real-time behavior. By this we mean no particular arbitrary operating speed, but simply that a practical system needs to respond according to the specific real-time demands of a problem environment. For example, an aircraft engine failure correction system really should do *something* before the vehicle impacts the ground. Finally, there is the constraint of operating within hardware constraints of the particular system environment. This may mean size and weight considerations in the case of flight systems, security considerations in such domains at air traffic control, or radiation hardening considerations in space-based systems.

4.2 Knowledge Acquisition, Representation, and Management

Aerospace vehicles, including commercial aircraft, high-performance fighters, space stations, or launch vehicles, are among the most complex devices ever designed and built by man. Knowledge-based systems, whether encoding experiential rules, qualitative structure-function models, or both, that attempt to contain comprehensive knowledge about those devices will be orders of magnitude larger than ones commonly in use today. Acquiring that knowledge and maintaining it in a reliable and useful form over multi-decade lifetimes imposes a substantial challenge on the AI field.

Three basic methods of knowledge representation have been developed by the artificial intelligence community [see articles in 4-1]. The first is commonly called *rules* because it sprang from the production rule framework of the cognitive psychology field. It stores knowledge in the form of: **IF *conditions* THEN *assertions***. Where the *assertions* can be additional facts for the knowledge base or actions to be taken when a set of facts is found. The second method is called *frame-based* knowledge representation, because the defining unit of knowledge is called a *frame* (sometimes a *unit* or an *object*). Each *frame* describes an entity in the knowledge base by means of properties

relating to that entity. The final method of knowledge representation is the formal scheme of the *predicate calculus*; this is often called *logic-based* knowledge representation.

Since the development of the first expert systems in the early 1970's (DENDRAL and MYCIN at Stanford), knowledge acquisition has always been the most time-consuming part of the process. Most expert systems have been built by hand-crafting heuristics (in the form of rules, frames, or expressions in the predicate calculus) as the result of a laborious interaction between domain experts and computer scientists (usually known as knowledge engineers). For many large-scale systems it has been estimated that each piece of experiential knowledge takes from an hour to a day to acquire, refine, and encode. In the late 1970's through the 1980's tools evolved to aid in the knowledge acquisition process. Such tools became known as *shells* and they provided convenient user interfaces, graphical displays, question-answering facilities, and the like. In some cases there was substantial success in removing the knowledge engineer as middleman and allowing domain experts themselves to encode their knowledge. However, even with the best of current tools, knowledge acquisition is still the dominant part of the expert system construction process. [see 4-2 and 4-3]

Consider the essential engineering knowledge that is involved in the design, building, test, and operations of a substantial aerospace device. In a study done by Ames Research Center, Marshall Space Flight Center, and Stanford University, it was estimated that, for the Hubble Space Telescope (HST), over 500 engineers had made a substantial contribution (i.e. worthy of including in a comprehensive HST knowledge base) during design and construction. This was focusing on the main telescope itself, excluding knowledge about the spacecraft and ancillary instruments. Moreover, those 500 engineers were scattered across at least a dozen physically distant sites in at least four different countries. Note that each of those 500 engineers communicates in at least a slightly different version of "engineering English," e.g. "high temperature" most likely has a slightly different meaning to each engineer. The current state-of-the-art in knowledge acquisition allows for the combination of knowledge from perhaps 3-5 disparate sources effectively; this means that future knowledge acquisition systems must improve by about two orders of magnitude in this dimension to be effective for realistic aerospace devices.

One promising approach to this bottleneck is to make knowledge acquisition a standard, "painless" part of the normal engineering design process. Research, at Stanford and NASA Ames Research Center, is underway on the development of an *electronic designer's notebook* (EDN) which replaces the traditional paper notebook of designers and integrates smoothly with the CAD/CAM tools that are now used to acquire formal quantitative knowledge in design. The EDN allows designers to communicate in the normal words and pictures of conceptual design and then "parses" those words and pictures into a formal knowledge representation language. This research has already shown success in acquiring heuristic conceptual design knowledge about the tertiary mirror assembly of the Space Infrared Telescope Facility and in allowing the knowledge to be used for design alternative analysis. [4-4]

A second research effort, focuses on developing an *interlingua* for knowledge bases. That would become a formal language for communication among large-scale systems. Developing a language that is rich enough to be useful for a wide variety of domains and at the same time usable within many different knowledge acquisition frameworks is a daunting task, but almost surely necessary for the kind of multi-site projects described above. [see 4-5]

Within Europe, a strong focus has been on studying knowledge acquisition in the context of knowledge utilization. A key development has been the KADS methodology, which defines four conceptual layers of knowledge relating structure of the knowledge base to functionality of the overall system. [see 4-6]. Several specific tools, most notably ProtoKEW and Muse, have arisen from this research.

Once knowledge has been acquired, validation and maintenance become important concerns for large-scale systems. Much has been written about the validation and verification of knowledge-based systems, but, to-date, the practical solution for that class of software is the same as for any other, namely extensive testing by simulation. Research is underway on such topics as automatic production of validated code from high-level specification, automatic production of software test suites, and automatic software fault analysis, but practical impact on a large scale is still in at least the medium-term future. Perhaps the best long-term solution for the knowledge base maintenance problem lies in the development of machine learning technology as described below. One can imagine knowledge base "introspection" systems constantly analyzing and testing, removing redundancies, correcting clear errors, and reporting inconsistencies to human experts.

Overall, the effort described in this section come under the topic of what has been called in the United States a "National Engineering Knowledge Base" project. The project points to the need for ten years worth of research and development on all of the issues of knowledge acquisition, sharing, testing, and maintenance described here. Such a project is still in its formative stages, but the potential for the aerospace community is so large, that the selection of an aerospace vehicle as at least one major test domain seems a wise choice.

4.3 Problem-Solving and Search

A major class of problems for knowledge-based systems are those where *combinatoric search* is the dominant paradigm. Here the task is to sort through an exponentially growing *tree* of possibilities which arises from starting from a given world state and applying *operators* to modify that state until a desired goal is reached. Since the

combinatorics of complete search of even a moderately complex problem-solving tree can tax the resources of the most powerful supercomputer, methods of *heuristic search* were developed to find satisfactory, if not necessarily optimal, solutions to problems of this class. See [4-7] for a detailed description of heuristic search methods. The most common types of tasks within the class of search problems are generically known as *planning* .

Planning is the process of developing a sequence of actions, that, when carried out in the world, will accomplish some set of possibly interacting goals. A sub-task of planning, usually thought of as an important problem in its own right, is scheduling: the placement of those actions into a precise timeline and the assignment of resources to each action. Taken together, planning and scheduling make up an important fraction of the decision-making that go into the control of any aerospace mission. Within the AI field, a great deal of work has occurred since the early 1970's in both areas. [4-8].

The scheduling problem is both conceptually simpler and computationally more intractable than the high-level planning task. Real world scheduling involve the interaction of hundreds, if not thousands, of actions and resources. While traditional computational methods, such as linear programming techniques, are important in scheduling, AI-based methods have proven important in at least three ways. First, the rich representational methods developed within the AI field are useful for acquiring and managing the complex constraints that govern many scheduling problems. In particular, frame-based knowledge representation systems provide an excellent semantics for describing constraints, and search-based constraint satisfaction techniques have proven their importance on a wide variety of scheduling tasks.

Second, in cases where schedule optimality is less important than schedule adequacy (as is the case in a majority of aerospace missions) AI-based *satisficing* methods, usually based on domain-dependent heuristics for guiding search, are capable of finding useful and correct schedules in far less time than conventional methods.

Finally, AI approaches have recently been applied to the vital task of reactive re-scheduling, where the goal is to produce a new schedule in response to changes in domain conditions or failures of the original schedule. Here, real-time solution production is vital (and the meaning of real time varies enormously between scheduling problems or even within a single scheduling domain). A technique known as *simulated annealing*, which allows for the continuous improvement of a complete, but perhaps inefficient schedule while maintaining schedule integrity at all time, is particularly applicable to the re-scheduling task.

Planning involves higher-level consideration of goals and actions within a problem domain. For some tasks, e.g. robotic path planning, the actions are fairly simple and combinatoric search is the dominant factor. For other planning tasks, however, e.g. tactical aircraft mission planning, the operations and their interactions with each other are very complex, and search combinatorics are not nearly as troublesome. Here, knowledge management to allow proper selection of actions from a modestly-sized collection becomes the major technical difficulty. Within the AI community, a wide variety of planning approaches have been researched. Two concepts that are particularly important are hierarchical planning and non-linear planning.

Hierarchical planning simply means removing details like operator preconditions and interactions from an initial selection of an action sequence. After the initial sequence has been chosen, details are added, perhaps in several steps, sometimes resulting in an addition of actions to correct troublesome interactions (in the simplest case where an action is found to destroy a necessary precondition for a later action), and sometimes resulting in backtracking at a more abstract level of detail to select a new action. Non-linear planning refers to the postponement of action ordering decisions until absolutely necessary.

Within the domain of aerospace systems guidance and control, AI-based planning and scheduling methods have wide applicability. Both aircraft and spacecraft missions are characterized by a need for both considerable pre-flight planning and an enormous amount of reactivity during the missions. Aerospace missions rarely go precisely according to plan. Sometimes changes need to be made for positive reasons (e.g. a Shuttle scientific experiment produced unexpectedly interesting data and crew wishes more time to repeat the experiment); often changes need to be made because of problems (e.g. a mechanical problem occurred on an aircraft and it took longer to get to a particular destination than expected). Four particularly interesting examples of aerospace mission planning and scheduling systems are worth mentioning.

The Pilot's Associate (PA) is envisioned as a pilot aiding system that performs many of the same functions currently accomplished by the Weapons Systems Officer (backseater) in two-man fighter aircraft. The PA is described in considerably more detail in Chapter VI, but one subsystem, the Tactics Planner will be discussed here.

The Tactics Planner subsystem reasons about threats and targets for attack or evasion, and tasks the Situation Assessment subsystem to monitor certain high interest tracks. The Tactics Planner does not invent tactics. It recommends tactics suitable to the actual situation, but selected from the pilot's own pre-briefed tactics or library of stored tactics. The selected tactic is then tuned to the geometry and timing of the existing situation. Approved tactics are monitored and adjusted or repealed and replaced in response to enemy countermoves.

The Tactics Planner maintains a set of viable response plans to situations that unfold as the mission progresses. It is based on a skeletal planning scheme in which a hierarchy of plan elements is maintained. Each plan element

carries its own knowledge base of when it should be invoked, how it is specialized, what other plan elements are related, actions that are necessary for execution, and the conditions of failure or completion. This data structure permits the system to support both sequential and simultaneous plans, and allows for multiple plan options to be considered and maintained as ready options until the situation dictates the selection of an exclusive option. The hierarchical structure permits the invocation of general plans even before precise details of the situation are known. Moreover, the distributed knowledge base allows a plan to be modified piecewise - at the plan element level - without destroying the larger structure of the plan. The Tactics Planner also includes a set of reflexive responses to urgent situations to support rapid response to previously undetected threats. [4-9]

A second system has been developed jointly by NASA Ames Research Center and Lockheed Space Operations Company. Here the task is to schedule the actions of hundreds of technicians who do ground processing of the Space Shuttle Orbiter at Kennedy Space Center. The traditional mode of doing business was for the "flow manager" to have one massive scheduling meeting per day with all sub-task leaders to produce a daily, immutable schedule on paper. The AI-based scheduler, called GPSS (Ground Processing Scheduling System), provides substantial assistance in producing that daily schedule, but, more importantly, gives the flow managers the freedom to reactively re-schedule during the entire three-shift day in response to actual progress in Orbiter operations. The iterative improvement mechanisms discussed above were of fundamental value in allowing such "anytime" re-scheduling to occur. GPSS is now in operational use for the Columbia and Endeavor Space Shuttle Orbiters. [4-10]

OPTIMUM-AIV is perhaps the best known scheduling tool in use within the ESA community. It was developed jointly by Matra Marconi Space, CRI, Progespace, and the AI Applications Institute. The system provides a variety of functions for spacecraft Assembly, Integration, and Validation (AIV), including process modeling, alternative schedule preparation and simulation, schedule conflict resolution, and schedule update verification. It is in current use for ARIANE IV equipment bay AIV activities and for SOHO satellite AIV. An important technical feature of the system is a sophisticated user knowledge entry front-end which allows for easy description of tasks, constraints, and scheduling strategies.

The MARS tool (Mission Activity and Resource Scheduling--developed by MBB-ERNO and in use at ESTEC), is in active use for space mission operations. Its tasks include analysis of Hermes operations, EURECA payload operations, Arianne production planning, and Columbus crew operations to mention only a few. MARS is a heuristic scheduler with an expressive task definition language, a sophisticated graphical user interface, and a rule-based, backtracking search mechanism. A derivative of MARS, NEPTUNE, has extended the basic methodology to coordinated, but geographically distributed scheduling problems.

Management of complex operations procedures (which one can think of as compiled and fully instantiated plans) is an important task for knowledge-based systems. Two examples of programs in current test use for automating the task of procedure execution in aerospace missions are PRS (Procedural Reasoning System) developed by SRI and in test use at Mission Control at Johnson Space Center, and EOA (Expert Operation Associate) developed by Matra Marconi Space and CRI, in test use at ESOC.

A final example is science scheduling for the Hubble Space Telescope. Here the task is to produce an observation timeline for the instrument, taking into account scientific and political priorities as well as physical constraints on the telescope. It is a highly-constrained, combinatoric task for which traditional, OR-based scheduling methodologies were shown to operate no better than 20 times real time on large computers. A heuristic scheduling system, called SPIKE, was developed by a team led by Mark Johnston at the Space Telescope Science Institute at Johns Hopkins University. This system, now operational, produces scientifically satisfactory schedules in real time on small workstation computers. [4-11]

4.4 Reasoning about Physical Systems

First generation AI-based systems rely on *heuristic* or experiential knowledge to solve problems like fault detection and diagnosis. "Rules of thumb," often represented in the form of simple procedural structures called rules, are used to match situations to actions that have proven diagnostic or corrective towards those situations in the past. These AI-systems, usually called *expert systems* have found many aerospace applications in recent years.

Expert systems are in routine use in both manned and unmanned space mission control at NASA. Within the Mission Control Center at Johnson Space Center, a series of diagnostic advisory systems, known collectively as Real Time Data Systems (RTDS), provide guidance to mission controllers in such areas as booster rockets, reaction control systems, communications, and fuel cells. RTDS is discussed in considerably more detail in Chapter VI of this report. At the Jet Propulsion Laboratory, a system called Spacecraft Health Automated Reasoning Prototype (SHARP) helped analyze problems with the communications system of the Voyager Spacecraft during the 1990 Neptune encounter. A later version of the system is operational for the Galileo Spacecraft power and propulsion subsystems. [4-12]

Within the ESA community, the best known operational expert system is Arianexpert, which provides post-flight telemetry data analysis for the Ariane launch vehicle. The systems combines AI and signal processing techniques along with advanced graphical and statistical capabilities. Within one major engineering domain, it has

reduced the effort required to provide post-flight analysis from 4 man/days to 1 man/day. Arianexpert is described in much greater detail in Chapter VI of this report.

Expert systems are only expert in their behavior when problems lie within the bounds of their pre-stored expertise. Beyond those bounds, they are extremely "brittle" providing no or (even worse) incorrect solutions. Human experts certainly use the experiential knowledge of expert systems for a major part of their problem solving, but are also capable of falling back on knowledge of device structure and functionality to do control and diagnosis when prior experience is not relevant. One may view this shift from experiential to *first principles* knowledge as moving from the job of a technician to one of an engineer. Since the early 1980's, research has been conducted in the AI field to develop the techniques necessary to model this form of reasoning and thereby build what might be called *2nd generation expert systems.* Two methodologies, in particular, have already shown considerable value in constructing these systems: *model-based reasoning* and *symbolic control.*

Model-based reasoning simply refers to the construction of formal descriptions of the structure of physical devices and of the relationship, through physical processes, of structure to function. Since, for devices above even moderate levels of complexity, it is impossible to completely describe the structure-function relationship by complete quantitative equations, a new field, known as *qualitative physics*, has evolved to allow reasonable predictions of behavior when such quantitative equations are unavailable. Through the use of model-based reasoning, it is now possible to diagnose device failures even when the failure has not been previously noted and encoded as an experiential heuristic.

Three model-based systems are of particular interest in aerospace domains. The first, from NASA Langley Research Center, focuses on the problem of jet engine diagnosis. The system, known as Faultfinder, encodes a qualitative model of turbojets at the level of major components (compressor, turbine, cowl, shaft, etc.). It is able to explain failures from symptoms from a comprehensive FAA database of commercial jet engine failures. An important feature is its ability to diagnose problems that resulted from physical, rather than explicit functional interactions. For example, it can explain hydraulic line punctures near engines which lost a turbine blade even though there is no predefined functional connection between those components. [4-13].

DIAMS2 is a model-based system developed by Matra Marconi Space for fault diagnosis of the Telecom 2 satellite. It incorporates both a behavioral and a functional model of the satellite to identify failures and suggest corrective procedures, with the behavioral model providing an initial global diagnosis and suggesting a specific component focus to the functional model. The system also provides a training capability for use by human satellite controllers.

A final system is the RCS diagnosis system built jointly by Rockwell Corporation and NASA Ames Research Center. This is part of the RTDS series of Space Shuttle monitoring and control systems discussed in Chapter IV. It maintains a detailed qualitative and quantitative model of the tanks, pipes, valves, pumps, and engines used for the small maneuvering jets of the Space Shuttle Orbiter.

There is wide potential applicability of AI-based physical systems reasoning technology to aerospace domains. Aircraft and spacecraft are exceedingly complex devices which are essentially impossible to model, in full detail, by conventional quantitative techniques. Many failures are discovered during testing and by simulation and can therefore be encoded within first-generation expert systems, but many more occur first during actual flight, and must be diagnosed and corrected in real time. Existing control methods work well where the precision and computational resources they require are available, but the real time and uncertain contextual demands of many missions point toward the need for symbolic control. The combination of accurate structure-function descriptions of aerospace vehicles along with qualitative reasoning methods should prove a vital adjunct to currently used techniques.

4.5 Control

Symbolic control methods are a complementary technology to model-based reasoning. Here the problem is not so much the lack of formal control methods for devices, but uncertainty in the environmental parameters that are required in formulating control laws when using the traditional control design methodologies. A technology known as *fuzzy logic*, originated by Professor Lotfi Zadeh of UC-Berkeley, allows reasoning about qualitative concepts like "high," "low," "moderately heavy," and the like within control law implementations. The majority of symbolic control work has gone on outside the aerospace domain to applications as diverse as camcorder focusing and subway acceleration. However, several aerospace companies in the United States, particularly Rockwell and MacDonnell Douglas, have begun explorations in fuzzy logic control of aircraft, and scientists at Johnson Space Center and Ames Research Center are beginning to apply the technology to spacecraft rendezvous and docking.

4.5.1 Flight Control

The design of flight control systems for modern, high-performance tactical fighter aircraft represent a significant challenges for control system designers. These challenges inhere in the unavoidable errors in modeling the complex, nonlinear dynamical behavior of high performance aircraft as well as in the uncertainties related to modeling and predicting exogenous disturbances. In many cases, a high performance airframe is achieved at the cost of aerodynamic instability, so that a very high rate active flight control system that is both fault and damage tolerant is

essential to flight safety. Because these aircraft rely on multiple effectors (e.g., ailerons, elevators, rudders, and possibly even canards and thrust-vectoring) and because the objective is to simultaneously control a number of outputs (e.g., position and orientation), the control system design problem is formally a multivariable one. Due to a combination of rigid-body and aerodynamic effects, the principal state variables associated with aircraft flight dynamics (altitude, velocity vector, orientation angles, and angular velocity vector) are coupled. As high performance combat aircraft continue to evolve, the flight envelope is likely to expand; indeed, the general trend is towards new flight regimes that are more complex and less well understood (e.g., post-stall maneuvering). Thus, trends for high performance aircraft that exacerbate the flight control design problem include:

- Flight envelope expansion — Nonlinear flight regimes
- Control effectors (e.g., canards, vectored thrust) — Higher dimensionality
- Relaxed-static-stability and agility — Faster control response and fault-tolerance needed

Over the past two decades advances in traditional system-theoretic control design methodologies have evolved to address many of these problems. Many of those methodologies and their application to flight control system design are described in the text by Lewis and Stevens [4-14] and are briefly outlined below. More recently, non-traditional control design approaches have evolved from the fields of fuzzy logic, reinforcement learning and intelligent systems. Two collections of works that describe recent developments in these fields are: *The Handbook of Intelligent Control* [4-15] and *An Introduction to Intelligent and Autonomous Control* [4-16].

Traditional Control Design Methodologies

Linear Quadratic Regulator (LQR) with *Loop Transfer Recovery* (LQG/LTR) designs shape the closed-loop multivariable frequency response to achieve good performance at low frequencies and robustness to unmodeled dynamics at high frequencies. [4-17].

Robust Control System Synthesis methods explicitly incorporate robustness in the design of a controller and are applicable to a class of linear time-invariant plants defined by a nominal linear plant model and a bounded range for uncertain model parameters [4-17]. In particular, H_∞ optimal control designs guarantee system stability robustness in the presence of specific types of modeling errors. The *μ-synthesis Design Methodology* mitigates some of the conservativeness inherent in H_∞ designs by mathematically searching for the least conservative design that satisfies robustness of both stability and performance.

Nonlinear Control Design approaches that directly address the nonlinear nature of the high performance aircraft plant include *sliding mode control* designs that track a desired trajectory in the presence of disturbances and parametric model uncertainties [4-18,4-19] and the *approximate input-state linearization* methodology that overcomes some of the shortfalls of traditional linearization approaches to nonlinear systems [4-20,4-21]. In the latter approach, any neglected nonlinearities and model uncertainties can be accommodated by designing the companion linear controller using a robust control system design methodology for the linearized model.

Adaptive Control System designs attempt to adjust on-line to accommodate unknown or changing system dynamics as well as unknown exogenous disturbances. The development of *robust adaptive* control techniques is the topic of ongoing research. There are two general classes of adaptive control laws [4-22]: ***direct*** and ***indirect***. *Indirect* adaptive control estimates system parameters on-line from a history of system inputs and outputs, and control law parameters are indirectly adjusted as a function of the estimated system parameters. In contrast, *direct* adaptive control law parameters are updated directly based on the history of system inputs and tracking errors.

Non-Traditional Control Design Methodologies

Learning Control Systems are similar to adaptive control systems in that control system parameters are adjusted on-line [4-15]. The differences between adaptive and learning control are essentially a matter of degree and emphasis. Adaptive control has a temporal emphasis: its objective is to maintain desired closed-loop behavior in the face of disturbances and dynamics that *appear to be* time-varying. In actuality, the changing dynamics are typically caused by unmodeled, nonlinear state-dependent (and therefore predictable) effects so that they are functions of state rather than of time. As a result, adaptive controllers must continuously re-adapt to compensate for this class of changing dynamics and this inefficiency results in degraded performance, since transient behavior due to parameter adjustment occurs whenever the dynamical behavior of the vehicle changes significantly. In contrast, learning augmented controllers exploit mechanisms that *associate,* a suitable control action or set of control system parameters with specific, measurable flight operating conditions. In this way, the effect of previously unknown nonlinearities can be anticipated and accounted for based on past experience. Once such a control system has "learned" transient behavior, there is potential for a greater efficiency and improved performance in comparison to purely adaptive control strategies. To accomplish this, learning control systems rely on general function approximation schemes that may be used, for example, to map the current operating condition to an appropriate set of control system parameters.

These schemes are based on neural network (or, more generally, connectionist systems) technologies. A brief overview of neural network fundamentals is presented in Appendix A of this report.

Novel hybrid control architectures that combine adaptation and learning in a synergistic manner are at the forefront of connectionist control system research and development. Adaptive approaches which address the problem of slowly time-varying and state-dependent dynamics and novel situations (e.g., those which have never before been experienced) are combined with learning approaches to accommodate nonlinear system dynamics.

The most common implementation of Fuzzy *Set Control* is essentially a rule-based control design approach that is appropriate for applications in which an accurate model is either not available or not required for acceptable performance [4-23]. Successes in a variety of applications, most notably consumer products, have redoubled interest in their application [4-24]. Fuzzy set theory generalizes traditional set theory to allow for partial membership in a set, allowing vague or imprecise information to be combined with precise information in forming a control law. Through fuzzy sets, linguistic variables are incorporated into control designs using fuzzy logic. A typical fuzzy logic statement that might be incorporated into an altitude control system might be:

If altitude_error is large and altitude_rate is small, ***then*** *thrust is medium*

Here, the terms *large, small* and *medium* all must be defined to have meaning in the context of set membership.

A control law is formed as a group of such fuzzy logic statements (rules). "De-fuzzification" is the process of producing a single, well defined control action by forming a weighted average of the outputs from each of the rules. The rules of a fuzzy set-based controller are formulated from an intuitive understanding of how the system behaves rather than from the more analytical methods of traditional, model-based control design approaches. Given a low number of inputs and outputs and a sufficient knowledge of the system dynamics, a fuzzy set controller can generally be designed quickly, will be easy to understand by others given the linguistic form of the rules and will be easy to adapt to changes in the system's performance objectives. Recently, approaches to control law design that combine learning and fuzzy rules have been developed wherein a subset of the fuzzy rules are created/learned based on a training set of observed, desired behavior and a complementary set is created base on physical intuition [4-25].

Intelligent Control approaches are intended to extend the range of autonomous or semi-autonomous operation achievable by a vehicle through onboard software that is able to monitor the vehicle's environment and resource usage, detect subsystem performance degradations and isolate loss of functional capability. In effect, these control laws are complete assess-"plan"-execute processes as described in Section 2.1 whose subfunctions are designed synergistically to create a complete "intelligent" controller. These capabilities are required in order to respond to abnormal events and maximize mission accomplishment. Throughout the coming decade these capabilities will become increasingly crucial to the development of unmanned autonomous aircraft for both military and civilian applications. Paraphrasing from the preface to [4-15]: "The fundamental goal of intelligent control is to fully utilize available knowledge of a system's behavior and real-time feedback regarding the system state and status to provide reliable control according to some predefined criteria (e.g., trajectory, objective function, goal achievement etc.) and to improve the capability of controlling the system over time through experience (i.e., learning through experience)." Current research is addressing the development of monitoring, diagnostic, reconfiguration and planning techniques along with suitable architectures to combine these techniques in a real-time system in order to optimize the level of vehicle/mission performance given its current operational status. Thus, the approach to describing, designing and implementing real-time work processes for G&C related application that is presented in this report is closely aligned with the goals of the intelligent control research community.

Stability is a safety of flight concern that must be addressed in the design of flight control systems. The major drawback of the application of the adaptive and non-traditional methodologies outlined above is the lack of analytical methods and the need for extensive full-scale testing in order to validate the stability of the designs given that they actively adjust the flight control system parameters on-line. Thus, the resolution of stability questions remains a significant impediment to the fielding of these classes of designs.

4.6 Machine Learning

Machine learning is the branch of artificial intelligence that focuses on the development of computational systems capable of improving their performance over time. As with humans, this improvement can arise by a number of mechanisms: by induction from trying to solve many example problems; by being taught; by analogy to other, potentially similar problems; and, by discovery and experimentation. This is a particularly dynamic field of artificial intelligence research at the present time, with many papers being published and small-scale systems being tested. In addition, several major groups are developing large-scale learning architectures, capable of learning in a wide variety of physical environments and problem-solving contexts. Any attempt to provide a comprehensive analysis of the field is well beyond the scope of this document; the interested reader is referred to [4-26,4-27,4-28]. Instead, we will provide several brief examples of current machine learning systems and then discuss potential applications to aerospace guidance and control. Note that machine learning methods can be applied to improve performance within all of the technology areas discussed above in this chapter.

An important step in building self-improving systems is being able to predict the future behavior of those systems from the data they produce (e.g. a telemetry stream from a spacecraft). An example of such a *data-driven* learning system is the AutoClass program developed at NASA Ames Research Center. AutoClass uses *Bayesian reasoning* to classify data from natural or man-made sources into most likely groups. Unlike traditional pattern recognition systems which need to be provided with a starting set of classes, AutoClass determines the most probabilistically likely set of classes (ranging from only one if the data is totally random). It has been tested on a wide variety of engineering and scientific databases (from Shuttle main engine exhaust plume spectra to DNA sequences). [4-29].

A second example, also from NASA Ames, applies a technique known as *explanation-based learning* to a system which helps schedule the Space Shuttle Orbiter's ground processing. Explanation-based learning simply means observing a failure in a traditional rule-based expert system and automatically creating and generalizing a rule based on that failure. In the case of the scheduling system, it means noting when a scheduling conflict is created by the system, "writing" a rule that prevents that specific failure, and then generalizing the pre-conditions of that rule using a knowledge-base of hierarchical scheduling constraints. These additional rules prevent the back-tracking that occurs during the normal search for solution of the expert system and thereby significantly speed up its execution over time. [4-30].

Neural or connectionist systems represent a third class of learning tools that can be applied to improve the performance in solving problems ranging from target identification and failure detection and identification to flight control and mission planning (see Appendix A). These systems rely on the ability of highly parameterized functions which have the potential for approximating any continuous function to an arbitrary precision. The key to their utility has been the development of simple "learning rules" from which the parameter values that are required to represent a given function are learned from repeated trials or presentations of known input-output pairs for that function (so-called *supervised learning*). For the examples cited above, the functions that are to be learned are:

Target Identification	Posterior probabilities as a function of extracted image features
Failure Detection and ID	Posterior probabilities as a function of extracted signal features
Flight Control	Vehicle dynamics as a function of observed state and measured control inputs (e.g., thrusts and aero-surface positions
Mission Planning	Plan search strategies as a function of the relative importance of, for example, time, fuel efficiency and survivability

The examples and a brief discussion of neural networks are discussed in Appendix A and Section 4.5. The research in the analysis, design and application of connectionist systems has been expanding at an apparently frenzied rate. A thorough overview of recent advances in supervised learning system can be found in [4-31]. Unsupervised learning systems for problems wherein "correct" output values are not known for a given input values have been developed for classification problems [4-32] and for optimization problems, such as control, for which the optimal solution must be learned through either real or simulated trials [4-33].

A fourth example is a method known as *genetic learning* where problem-solving procedures are developed by a process analogous to evolution of biological systems. The Naval Center for Applied Research in AI has built a program capable of learning optimal missile avoidance behavior on simulated test problems. The program starts with a complete set of primitive operations for an airplane (speeding up, slowing down, banking left and right, etc.) and a simple initial avoidance procedure (e.g. bank left and slow down). Many simulated missiles are fired at the airplane. Whenever it successfully avoids the missile the current avoidance procedure is reinforced; whenever it is struck by the missile (almost all the time initially), the procedure is mutated by removing, deleting, or changing an operation (favoring retention of those operations which were part of many successful avoidances). Eventually, a procedure results that cannot be improved further. [see 4-34 for more information on genetic learning]

A final example is the SOAR architecture originally developed by Allan Newell, John Laird, and Paul Rosenbloom when all were at Carnegie-Mellon University. SOAR is a general-purpose learning architecture based upon the paradigm of state-space search. Its framework encompasses any computational system utilizing combinatoric search. SOAR stores useful search paths as *chunks* which, when deemed useful on future problems, replace the paths in the search tree. Gradually, search is replaced by direct retrieval of successful solutions. SOAR has been applied to many search problems ranging from computer configuration to robotic path planning. [4-35]

The potential applicability of machine learning to the guidance and control of aerospace vehicles is as broad as the applicability of computational systems themselves. Aerospace vehicles and missions are complex and operate in uncertain and changing environments. In general, it is impossible to predict all behaviors and failures ahead of time, and therefore impossible to build diagnostic, planning, or other problem-solving systems capable of successful

operation in all situations. Even if it were possible to achieve completeness initially in a problem-solving system, the vehicles themselves tend to be long-lived (e.g. 30 + years lifetime predicted for NASA's Space Station Freedom) with many components changing over time. Machine learning does not in itself solve any problems, but it is capable of making any problem-solving system behave more robustly and efficiently.

4.7 Understanding the External Environment

Image understanding is the process of taking an input signal form in some wave band and extracting information relevant to some task. Most image understanding work has gone on in the visual and IR spectra, but the process description is equally applicable to acoustic image understanding, such as that seen in acoustic structures analysis. The problem is difficult because the inputs are noisy, the typical sensors (camera or IR) have already reduced a 3D scene to 2D, the objects of interest are hard to define (what defines a target of opportunity?), and interesting applications need to operate in soft real-time. Since the problem of image understanding with no other interpretive aid is so difficult, image understanding is rarely applied without some other dynamic data source, such as inertial navigation sensors (INS) or global positioning system (GPS) sensors for navigation, or laser or optical range finder systems for object recognition in complex environments. This added information eases the image understanding task, at the cost of introducing multi-sensor data fusion issues (see below).

The process involves at least the following steps:

- digitization or sampling - forming discrete samples of image intensity across the input signal form
- smoothing - eliminating spurious noise
- segmentation - grouping the image elements into related sets
- labeling - providing potentially redundant reference tags for the segments
- analysis - grouping the labeled segments to form coherent objects

The process steps, analogous to the problem of speech understanding, are not cleanly separable. For example, connected components analysis may suggest that the initial image segmentation was too aggressive, and the segmentation repeated to form fewer elements. The number of iterations required can be reduced by simplifying the problem as follows:

- region of interest identification - an external system (camera-pointing system) identifies the area of the environment that will contain the object(s) of interest.
- environment simplification - object obfuscation (fog, specular reflectance, etc.) is eliminated, obscured objects are minimized, and so forth.
- limited object set - the number of types of objects that must be identified is reduced to increase the inter-class discriminability

The two primary application areas for image understanding have been object recognition, (e.g., automatic target recognition (ATR)), and navigation aiding (e.g., passive ranging). Examples include passive navigation for rotorcraft flying nap-of-the-earth (NOE) missions [4-36], autonomous aircraft landing [4-37], and object tracking from mobile platforms [4-38].

Multi-sensor data fusion is the process whereby reports available from a variety of sensors are combined to form a single, cohesive view of a situation which is better than any single sensor system could provide in terms of robustness, reliability, and fidelity of the view. As an example, a millimeter-wave (MMW)/ forward looking infrared (FLIR) combination, applying a range normalization to pixel measurement, can derive a physical measurement of the detected object. Another example is the creation of a new feature, e.g. volume, from two features extracted from two sensors, e.g. area from FLIR, and Length from MMW.

Multi-sensor data fusion has several technical challenges. Most come from the vast disparity the sensor reports arriving at the fusion processor (either human or computer). These reports can vary in a number of dimensions including: coverage area, temporal characteristics of coverage, field of view, angle of view, range, resolution, update rate, detection probability, modality of report (audio, electro-optic, magnetic, ...), degree of complexity/realism of report, type of target information, and the temporal characteristics of the reports. Multi-sensor data fusion occurs at different hierarchical levels, e.g. at the pixel level, object level, feature level, decision level, and so forth. Theoretically, this fusion can occur at any level. In practice, however, fusion takes place at any level after regions of interest, i.e. objects, are detected. The selection of the appropriate level(s) at which to perform fusion involves a tradeoff: fusion at the lower levels demands more computation, but operates on more complete data before any raw information is lost.

Data fusion occurs in conjunction with traditional signal processing. Sensor fusion processes data from a multiple sensor suite so as to exploit their synergy and improve the robustness and reliability of the situation

representation. Understanding the characteristics and phenomenology of the sensor suite is crucial in sensor fusion. The goal is to recognize the strength and weakness of each sensor and the complementary nature of the information provided by the sensors. This knowledge is used to formulate a strategy or rule based inference to direct the fusion process.

Since a common feature of any sensor system is that the data is noisy, and thus to some degree unreliable, a universal problem in trying to combine data from disparate sensor systems is to decide how much confidence one should have in the inputs of one sensor system with respect to another. One approach to this problem is to use evidential reasoning. Evidential reasoning methods include weighting and majority approaches, confidence factors (such as those used in MYCIN), fuzzy logic, Bayes nets, and the Dempster-Schafer rule of combination. The weighting and majority rule is an ad hoc suboptimal solution, but is simple and easy to implement. Confidence factors, mostly used in medical applications, suffer a major drawback which is arbitrary assignment of the belief and disbelief by an expert. Fuzzy logic, an area of focused research attention, is a well developed theory but has not yet been significantly exploited in sensor fusion. Bayes nets have become very popular in recent years; they are based on Bayesian probability and a class of networks typified by the use of "influence diagrams" in decision analysis. The Dempster-Schafer rule of combination provides a means for assigning and manipulation of mass distribution functions.

Data fusion approaches are found throughout the aerospace industry. The Enhanced Situation Awareness System (ESAS) system, a current research effort undertaken by a consortium of avionics, radar and airframe suppliers, provides an out the window heads up view synthesized from inputs taken from a variety of sensors, allowing a flight crew to perform takeoff and operations in and around what would otherwise be untenable meteorological conditions, including windshear, wake vortex, and fog conditions. Other examples include INS/vision passive ranging [4-39], [4-40] and GPS/INS for autoland [4-41].

4.8 Human-Machine Interaction

The goal of human-machine interaction technology development is to increase the bandwidth and decrease the error rate in human-computer interfaces. Obvious approaches to these goals include:

- adapting the machine interface so that human inputs take the form of more natural and commonplace actions (picking up an iconic or even virtual folder and dropping it in the appropriate location, as opposed to keying, "mv foo bar".)
- adapting the machine's outputs to common human experience (watching a wing flex, as opposed to tracking a scrolling column of stress values)
- adapting the machine's reasoning process so that explanation of its intermediate states is possible

The first two of these approaches are taken to the limits of available technology in virtual reality. Virtual reality aims to overcome the limitations of conventional human-computer interaction by matching the computer much more closely to the capabilities of human sensorimotor systems. The goal is to improve computer hardware and software in order to match human capabilities and requirements, rather than to downgrade human performance to match the computer's limitations

The components of a virtual reality system include a computerized description of the environment to be studied or manipulated, stereo 3-D viewing system, 3-D auditory system, and data input device(s). The user may have the capability to interact with the database through gloves, gesture commands, voice input/output or other aids for computer interfacing. Body motion tracking devices and artificial motion-inducing techniques may also be utilized. In other words, virtual reality is a display and control concept, which allows a human, wearing or using special devices, to virtually experience a remote, actual environment or one fabricated by the computer from a database. This system has the potential to create an illusion so compelling and complete that it appears real to the observer.

A number of organizations in the United States are sponsoring research on potential applications of virtual reality. The National Science Foundation, Office of Naval Research and the Naval Research Laboratory are supporting basic research efforts including development of the principles of multimedia human-computer interaction for scientific visualization, potential applications of virtual reality to spatial reasoning, software for data integration, interactive visualization of abstract data of high dimensionality and potential utilization of virtual reality in learning and training. [See 4-42 for more information on virtual environments].

The Human Interface Technology Laboratory at the University of Washington focuses on a development of special class of software programming which takes into account the perceptual organization of the human and dynamically creates the three-dimensional sound, video, and tactile images which surround the user. The Laboratory has formed a consortium of twelve companies who have a number of developments in various stages of progress. These include:

- A retinal microscanner that will use lasers to directly scan images onto a viewers retina

- An electronic wand or three dimensional input device for point-and-speak actions in the virtual environment;
- The Virtuphone, a video and audio input/output device to replace the telephone and TV in the home.

NASA Ames is currently investigating the underlying perceptual principles of auditory displays which have the ability to generate localized sound cues in a flexible and dynamic manner and is also developing a prototype signal processor based on these principles. The prototype maximizes portability by synthetically generating 3-D sound cues in real-time for delivery through earphones. Unlike conventional stereo, sources will be perceived outside the head at discrete distances and directions from the listener.

Other countries, such as Great Britain and Japan also have the beginnings of an industrial capability, but the United States is the principal developer and user of the technology.

There are some major technology issues which remain to be addressed. These include:

Visualization and spatial interpretation of massive databases, such as a planetary/lunar surfaces, earth terrain/air traffic control data, the human body, ocean-land interfaces, etc.

Interactive presence to manipulate, recombine or restructure complex, environmental data sets (real or hypothetical).

Generation of visual, aural, and tactile representations via information processing subsystems.

Degree of sensory distortion, imaging limitations and information representations congruent to normative human behavior.

Advanced visual, aural and kinesthetic rendering and feedback capabilities for analysis of the real or hypothetical database.

Advanced computer science for data access and processing and multi-sensory fusion algorithms.

A high resolution, wide angle, true color image delivery system for each eye.

The ability to track the user's head position (and other parts of the body) within a millimeter and orientation to a fraction of a degree. The lag between head position and visual field cannot be greater than 50 milliseconds. Lag is the biggest problem today, along with a narrow, 60 degree field of view.

Though considered a part of many virtual reality systems, speech understanding is particularly noteworthy as a natural human interaction mechanism which has been particularly difficult to realize in human-machine interactions. Speech understanding combines two technologies which are by themselves challenging, namely speech recognition and natural language understanding. Speech recognition is the process of taking a time-varying acoustic signal (speech) and extracting the phonemes or individual sound units, then composing them to form words. This process is a special case of the image understanding problem discussed earlier. Speech understanding is a interdependent layer above this which extracts task specific meaning from the words.

To illustrate the interdependence, in (connected) speech recognition it would be difficult if not impossible to distinguish "pilot scare" from "pilot's care" without the context in which the phrase occurred. Similarly, the phrase, "It's time" could be an interrogative, exclamation, or statement carrying varying levels of stress, certainty, and so forth. Without reference to the intonation (present during the speech recognition), it would be difficult to determine which of those forms was intended. Other simplifications are typically made to reduce the degree of interdependence and the knowledge which must be brought to bear on the problem. Common simplifications are:

Isolated Word Recognition - if the speaker carefully enunciates "pilot" <pause> "scare", the complexity of context dependent interpretation is avoided.

Speaker Dependence - if the recognition system can be trained to the way that a single speaker says "four", then the bounds on what the system will accept as an occurrence of "four" can be tightened, resulting in a lower error rate (higher accuracy).

Limited Vocabulary - if the only possible words are "tune", "first", "second", "increase", and "reduce" the possibility for word confusion is reduced, and accuracy increases. This can often be structured by means of a finite state grammar that permits only a certain subset of the vocabulary to be recognized, given the preceding utterance.

For example, the October 1990 Space Shuttle mission, STS-41, carried an experimental voice recognition system for control of four payload bay cameras which was based on a speaker dependent, 41 isolated word/phrase vocabulary. With on-mission retraining of the templates, 100% accuracy was achieved for one speaker over a three day period [4-

43]. Honeywell has demonstrated dynamic retraining for a similar application (radio tuning in military transport) which eliminates the need for using a separate retraining mode during operations [4-44]. The European Fighter Aircraft has a demonstrated a system with up to 1000-word vocabulary, achieving 99% accuracy on isolated digit recognition in 7dB peak signal-to-noise ratio conditions [4-45].

In less challenging environments (stress-free, single task, no noise, fixed head position), some of these restrictions can be relaxed: current laboratory systems can achieve 99+% accuracy on speaker independent connected speech for a small vocabulary [4-46], and office systems can recognize 5-10,000 word vocabularies for speaker dependent isolated word recognition with correction.

4.9 References

[4-1] Friedland, P., "Special Section on Architectures for Knowledge-Based Systems," *Communications of the ACM*, Vol. 28, No. 9, pp. 900-941, Sept. 1985.

[4-2] Boose, J. and Gaines, B., "Knowledge Acquisition Tools for Expert Systems," Academic Press, London, England, 1988.

[4-3] Gruber, T., "The Acquisition of Strategic Knowledge," Academic Press, San Diego, CA, 1989.

[4-4} Baudin C., and Gevins, J. "Using Device Models to Facilitate the Retrieval of Multimedia Design Information," to appear in Proceedings of International Joint Conference on Artificial Intelligence-1993, Chambery, France, 1993.

[4-5] R. Patil, R. Fikes, P. Patel-Schneider, D. Mckay, T. Finin, T. Gruber, R. Neches; The DARPA Knowledge Sharing Effort: a Progress Report; in Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning; Cambridge, Massachusetts; October 25-29, 1992.

[4-6] Anjewlerden, A., Wielinga, B., and Shadbolt, N., "Supporting Knowledge Acquisition: The ACKnowledge Project," in ESPRIT-92 Knowledge Engineering.

[4-7] Barr, A. and Feigenbaum, E, "The Handbook of Artificial Intelligence," William Kaufmann, Inc., Los Altos, CA, 1981.

[4-8] Wilkins, D., Practical Planning: Extending the Classical AI Planning Paradigm. Morgan-Kaufmann, San Mateo, CA, 1988.

[4-9] Lafferty, Larry , G. Edwards, M. Hoffman and J. Shoop. "A Real Time Planner for Tactical Air Combat," in Proceedings of the AIAA Computing in Aerospace Conference 8, Baltimore, Maryland, October 1989

[4-10] Zweben, M., Davis, E., Daun, B., and Deale, M., "Heuristics versus Lookahead in Iterative Repair Scheduling," to appear in Proceedings of International Joint Conference on Artificial Intelligence-93, Chambery, France, 1993.

[4-11] Minton, S., Johnston, M., Philips, A., and Laird, P., "Solving Large-Scale Constraint Satisfaction and Scheduling Problems Using a Heuristic Repair Method," in Proceeding of Eight National Conference on Artificial Intelligence, Boston, MA, pp. 17-24, 1991.

[4-12] Atkinson, D. and Mark, J., "Applications of AI for Automated Monitoring: The SHARP System," in Proceedings of the Second International Symposium on Space Information Systems, Pasadena, CA, Sept. 1990.

[4-13] Abbott, K., "Robust Operative Diagnosis as Problem-Solving in a Hypothesis Space," in Proceedings of Seventh National Conference on Artificial Intelligence, Saint Paul, MN, pp. 369-374, 1988.

[4-14] Lewis, F. & Stevens, B. (1992). *Aircraft Control and Simulation*, John Wiley and Sons.

[4-15] White, David and Sofge, Donald (eds.), *Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches,* Van Nostrand Reinhold, 1992.

[4-16] Antsaklis, P.J. and Passino, K.M., (eds.), *An Introduction to Intelligent and Autonomous Control,* Kluwer Academic Publishers, Boston/Dordrecht/London, 1992.

[4-17] Doyle, J.C. and Stein, G., "Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis, *IEEE Transactions on Automatic Control,* AC-26, pp. 4-16, 1981.

[4-18] Skowronski, J.M., *Control of Nonlinear Mechanical Systems*, Plenum Press, New York, 1991.

[4-19] Utkin, V.I., *Sliding Modes and Their Application in Variable Structure Systems*, MIR, Moscow, 1978.

[4-20] Su, R., "On the Linear Equivalence of Nonlinear Systems", *Systems and Control Letters*, Vol. 2, No. 1, July 1982, pp. 48 - 52.

[4-21] Krener, A.J., S. Karahan, M. Hubbard and R. Frezza, "Higher Order Linear Approximations to Nonlinear Control Systems", *Proceedings of the IEEE Conference on Decision and Control*, Los Angeles, 1987, pp. 519-523.

[4-22] Åström, K. and B. Wittenmark, *Adaptive Control*, Addison-Wesley, 1989.

[4-23] Tong, R.M., "A Control Engineering Review of Fuzzy Systems," *Automatica*, Vol. 13, pp. 559-569, 1977.

[4-24] Lee, C.C., "Fuzzy Logic in Control Systems: Fuzzy Logic Controller Parts I and II," *IEEE Transactions on Systems Man and Cybernetics*, Vol.-20, No. 2, March/April, 1990.

[4-25] Wang, L.X. and Mendel, J.M., "Generating Fuzzy Rules by Learning Through Examples," *IEEE Trans. on Systems, Man and Cybernetics*, Vol. -22, No. 6, Nov./Dec. 1993.

[4-26] Michalski, R., Carbonell, J., and Mitchell, T. "Machine Learning: An Artificial Intelligence Approach, Volume I," Morgan Kaufmann, San Matgero, CA, 1983.

[4-27] Michalski, R., Carbonell, J., and Mitchell, T. "Machine Learning: An Artificial Intelligence Approach, Volume II," Morgan Kaufmann, San Matgero, CA, 1986.

[4-28] Kodratoff, Y., and Michalski, R., "Machine Learning: An Artificial Intelligence Approach, Volume III," Morgan Kaufmann, San Matgero, CA, 1990.

[4-29] Cheeseman, P., et. al., "Bayesian Classification," in Proceedings of Seventh National Conference on Artificial Intelligence, St. Paul, MN, pp. 607-611, 1988.

[4-30] Eskey, M. and Zweben, M., "Learning Search Control for Constraint-Based Scheduling," in Procedings of Eighth National Conference on Artificial Intelligence, Boston, MA, pp. 908-915, 1990.

[4-31] Hush, Don R. and Horne, Bill G., "Progress in Supervised Neural Networks," *IEEE Signal Processing Magazine*, Vol. 10, No. 1, January 1993.

[4-32] Carpenter, Gail A. and Grossberg, S, eds., "Pattern Recognition by Self-Organizing Neural Networks," MIT Press, Cambridge, MA, 1991.

[4-33] Barto, A., Sutton, R., and Anderson, C., "Neuronlike Adaptive Elements That Can Solve Difficult Control Problems," IEEE Transactions on Systems, Man and Cybernetics, Vol. 13, No. 5, 1983.

[4-34] Rawling, G., ed., *Foundations of Genetic Algorithms*, Morgan-Kaufmann, San Mateo, CA, 1991.

[4-35] Laird, J., Newell, A., and Rosenbloom, P., "SOAR: An Architecture for General Intelligence," Artificial Intelligence, Vol. 33, pp. 1-64, 1987.

[4-36] Menon, P. K. A. and Sridhar, B., "Passive Navigation Using Image Irradiance Tracking," pp. 639-646, AIAA Guidance, Navigation and Control Conference, Boston, MA, Aug. 14-16, 1989

[4-37] Dickmans, E.D., and Graefe, V. "Dynamic Monocular Vision and Applications of Dynamic Monocular Machine Vision", AGARD GCP Symposium on Advances in Techniques and Technologies for Air Vehicle Navigation and Guidance, AGARD-CP-455 (This is Ray's reference).

[4-38] Bhanu, B. and Burger, W. "Qualitative Motion Detection and Tracking of Targets from a Mobile Platform", pp 289-318, Proceedings DARPA Image Understanding Workshop, Cambridge, MA, April 6-8, 1988.

[4-39] Roberts, B. and Bhanu, B. "Inertial Navigation Sensor Integrated Motion Analysis for Automomous Vehicle Navigation", pp 364-375, Proceedings DARPA Image Understanding Workshop, September, 1990.

[4-40] Roberts, B.A., Sridhar, B. and Bhanu, B. "Inertial Navigation Sensor Integrated Motion Analysis for Obstacle Detection," pp 131-136, Proceedings 1991 Digital Avionics Systems Conference.

[4-41] Snyder, S., Schipper, B., Vallot, L., Parker, N., and Spitzer, C. "Differential GPS/Inertial Navigation Approach/Landing Flight Test Results", IEEE Position Location and Navigation Symposium, Monterey, CA, March, 1992.

[4-42] Ellis, S. "Nature and Origins of Virtual Environments: A Bibliographical Essay", *Computer Systems in Engineering* , V2, No 4, pp.321-347, 1991.

[4-43] Salazar, G. "Voice Recognition Makes Its Debut on the NASA STS-41 Mission," Speech Technology Vol. 5, No. 3, pp. 86-92, Oct-Nov. 1991.

[4-44] North, R. and Graffunder, K. "Dynamic Retraining Approaches for an Airborne Speech Recognition System," NAECON 85, pp.970-974, Dayton, Ohio, May 20-24, 1985.

[4-45] Galletti, I. and Abbott, M. "Development of an Advanced Airborne Speech Recognizer for Direct Voice Input", Speech Technology Vol. 5, No. 1, pp. 60-63, Oct-Nov. 1989.

[4-46] Levin, E. "Connected Word Recognition Using Hidden Control Neural Architecture," Speech Technology Vol. 5, No. 3, pp. 102-107, Oct-Nov. 1991.

CHAPTER 5

INFRASTRUCTURE SUPPORTING TECHNOLOGIES: DEVELOPMENT ENVIRONMENT FOR KNOWLEDGE-BASED SYSTEMS IN GUIDANCE AND CONTROL

5.1 Introduction

Knowledge based system (KBS) development requires capturing knowledge (elicited from experts or reports) and transferring it into a computer formalism. This transfer is risky and is very time consuming. Due to certain characteristics which invalidate the classic waterfall development cycle (see 2.3.1), KBS development requires new methodologies.

This chapter explores existing methodologies and tools for developing such a system and analyses their adequacies for the G&C world. It is not intended to be a comprehensive study of commercial products dealing with this problem; it instead tries to explore functionalities existing in some tools that may be useful (from a productivity point of view) during the KBS development phase.

The remainder of this chapter is organized into three sections. In the first section, particular problems associated with the use of KBS in G&C are identified. The second section describes some software engineering results. One characteristic of KBS lies in knowledge acquisition and modeling; methods dealing with this problem are also explored. The third section of this chapter describes features of tools that may be useful during the KBS development. Software, hardware and integration aspects are considered.

5.2 Characteristics Of KBS Development In G&C

The complexity of problems in the aeronautics field led designers to use KBS as an easier and more comfortable programming solution than classical laborious and error prone software development. Soon design tools, such as KEE and ART, built on top of AI languages like Lisp or Prolog became available. With those software tools, developers could quickly produce promising prototypes in advance of production systems delivery. No formal methods were applied and it soon became apparent that high quality maintainable systems were not being produced. Since then, a disillusionment has been felt in the AI community. Two lessons have been learned from these experiences:

- KBS applications required the development of dedicated methodologies because of certain specificities which modify the development cycle ;
- KBS tools should be carefully selected for the appropriate domain so that the KBS application can demonstrate better performance than classical procedural type software.

These practical considerations led to consideration of specific methodologies. The characteristics of KBS development for G&C applications are surveyed in this chapter, with less than an exhaustive coverage, in order to introduce the second part of the chapter where existing methods and tools are analyzed. For example, the validation and verification concerns are not covered (see section 6.1.4 for one successful approach to this problem). Problems associated with the inherent need for capturing human expertise are first surveyed. Then the results shown in the previous chapters will be used for identifying important points in the G&C domain.

5.2.1 Difficulties In Capturing Human Expertise

To understand KBS potential and complexity of development, one has to take into account the importance of human expertise in the design process. Human domain experts need to be considered as full members of the project team and involved at each stage of the development, e. g. early description of the problem domain, requirements definition, design, description of performed tasks, ideas of new man-machine dialogue, validation, end-use, etc. This central role influences the developing environment and suggests modification of the infrastructure. It can also be mentioned that KBS attempts to standardize the knowledge representation in order that human specialists can more easily understand what is in the machine. It requires the use of new concepts like objects, plans, heuristics, agents, etc. The expert may even want to change quickly from one representation to another. Consequently, knowledge engineers have to walk their way through a very large set of representation schemes.

Another problem with human experts is that they frequently change their minds or improve the precision of their knowledge. The design team is faced with a very iterative knowledge acquisition process resulting in frequent

changes to the knowledge representation in the system architecture. This unstructured, imprecise, incomplete, and evolutionary nature of human expertise creates new needs for mastering the quality and validity of the KBS design. It may well be the main obstacle to the operational use of KBS technology.

5.2.2 What Is Specific With KBS Used In Guidance And Control ?

Knowledge handling in G&C applications is to some extent atypical. A functional analysis of knowledge based G&C systems is given in chapter 2.1. Specific examples of G&C KBS applications are discussed in Chapter 6 and illustrate many G&C unique characteristics. A G&C KBS is frequently a real-time system, incorporating some temporal reasoning and having constraints due to its integration into a vehicle. Indeed with conventional systems, the operator is often out-of-the-loop and the reaction time is on the order of one second (faster reactions are treated by fully automated systems). With automation aided systems, the user is in-the-loop and, due to this much stronger interaction, the reaction time is within tens of seconds. Others levels exist. They may be characterized by the number of involved agents as in air-traffic control or battlefield management. Response times are then in the order of minutes or hours.

G&C applications typically involve a two loop architecture with a very reactive inner control loop (monitoring - diagnosis - planning - decision making - implementation) and a time dependent, but less constrained, outer guidance loop (supervision - coordination - external agent(s)). The transverse description of decision systems proposes three stages across these loops for assessment, planning and execution. The world in which these decision-oriented KBS's are supposed to work is complex and applies to a large variety of domains (avionics supervision, pilot assistance, tactics and maneuvers, air-traffic control, battlefield management, etc.). A very important consequence is that all these systems are multidisciplinary in nature, mixing knowledge from various sources and not mono-expert, mono inference type of systems.

A second observation is that time constraints are always severe in G&C. Even if strict real time operation can be avoided, as in many pilot's assistance applications, at least human-perceived real time must be addressed. Real time does not mean fast, but means faster than the supervised process (and consideration of implemented functions). Such a feature implies that the KBS must be able to reason under time constraints (coming up with a solution within a defined time), monitor asynchronous events, perform continuous operations and be efficiently coupled with some numerical components solving various real-time problems (such as planning, diagnosis, resource monitoring, signal processing and analysis, etc.). This remark concerns not only performance aspects but, more fundamentally, also addresses the capacity to deal with time in expressing and exploiting knowledge and in putting constraints on the strategies so that solutions can be reached in the time available. Temporal reasoning is non-monotonic; data are either not durable or decay in validity with time. The fact that the variable "time" differs drastically from others (it constantly varies) has consequences at several levels. Such a system must be able to handle time and reason with dated facts (about past, present, and future events as well as the sequence in which the events occur). It must be able to cope, to a certain extent, with missing and uncertain data. Very few KBS prototypes have faced these types of requirements so far and methodologies and tools have largely ignored this problem.

Finally, avionics systems are highly integrated systems and neither the common black box approach nor are loosely coupled module approach are likely be the appropriate answer for the designer of KBS's in this environment. Implementation of the final version of the KBS into a vehicle implies that it has to fit within limited computing resources, be highly integrated into the vehicle environment, and be reliable and predictable in the way it processes data and interacts with the environment. In such real world applications, designers cannot put aside issues associated with the integration of the KBS with conventional systems and with verification and validation constraints.

5.3 Methodologies To Support The Knowledge Engineering Life Cycle

As shown in chapter 2.3, the life cycle characteristics of KBS differ in some ways from the traditional software engineering life cycle. Two main methodologies were mentioned, prototyping and waterfall development. To support such a life cycle, methods have to be adapted to these activities. Rapid prototyping and conventional software development are methodologies presently in use. Some examination of their advantages and drawbacks are given here and a possible approach taking advantage of them both is presented.

5.3.1. Rapid Prototyping

The use of rapid prototyping to build a complete system using one of the many shell packages available on the market is the most common method of developing a KBS. This method, also known as "evolutionary prototyping" or "iterative prototyping", was deduced from experimental approaches to KBS development. It consists of iterating the cycle :

- expertise collection,

- implementation - knowledge modeling & coding with the KBS development tool,
- validation and test with the expert(s),

until there is no more knowledge to capture. This development approach stresses :

1. early and constant expert involvement ;
2. direct expert interaction with software engineers ;
3. structured rapid prototyping ;
4. incremental validation procedures (as soon as possible in the application life cycle).

A possible representation of the life cycle associated with this methodology is shown in figure 5.1. This methodology does not incorporate large, sequential analysis, design and implementation phases resulting in a fully developed system. Rather, it leads to the delivery of a rapidly constructed prototype system to the client at an early stage of development (the requirements and modelization phases are not very time consuming). The prototype then dynamically evolves as further expertise or requirements are fed back to the development team. The process therefore involves a gradual refinement of an initially crude solution into the desired one.

Rapid prototyping of the developing system serves several useful and important purposes. First it provides both the expert and the engineer a substantial, "visible" version of the emerging system. Thus, the prototype is a common reference point that grounds communications between experts and engineers and reduces ambiguities and misconceptions. The prototype is also a useful testbed that allows the expert to inspect, criticize, refine and incrementally validate implemented knowledge. This process ensures steady progress towards useful functionality and enthusiastic user acceptance at the conclusion of the program. In a similar vein, the prototype allows the user to better envision the final version of the system in an operational environment and thus suggest reasonable refinements of functional requirements at appropriate places in the development cycle. The result of this insight is the final delivery of a truly operational version of the system. Finally, rapid prototyping may provide important program management insight into the progress of the development effort and thus reduce the risk of ultimate failure.

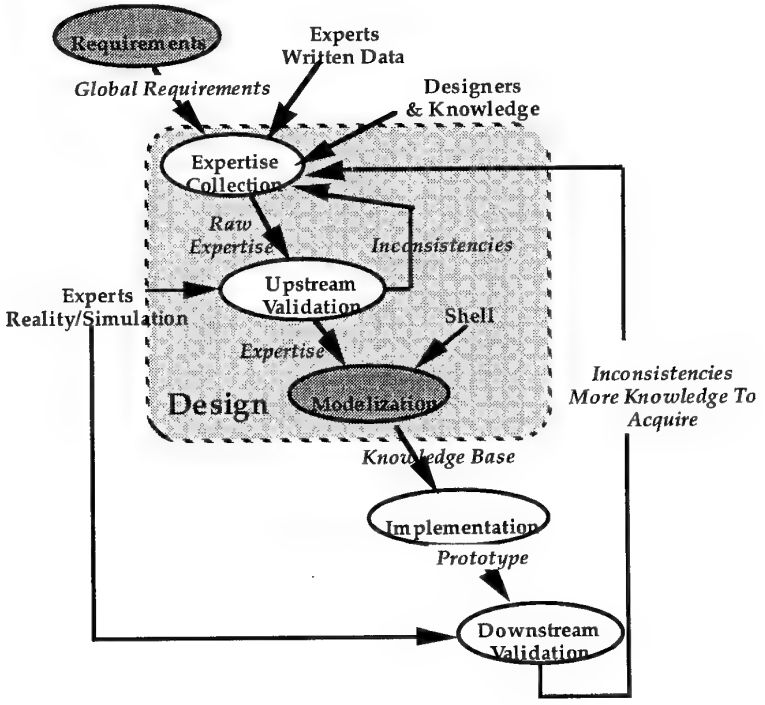


Figure 5.1 A Rapid Prototyping Life-Cycle Model

On the other hand, development of prototypes does not use any underlying life-cycle model to guide and control. The direct coupling between specification, design and implementation phases means that each iteration involves re-specification, re-design, re-implementation and re-evaluation activities. Moreover, major design decisions are also made very early in the development process. These may very well be totally inappropriate but are nevertheless locked into the system. These inappropriate decisions may not be discovered until later, whereupon the system design is radically altered or the design is scrapped and re-started which leads to delays, wasted resources and extra costs. However, these added costs and delays are far less than what would have been experienced in a classical software development process. Analysis is implementation-dependent and driven by the prototype. The knowledge acquisition is driven and constrained by the fixed, pre-defined implementation formalisms and inference mechanisms imposed by the shell. Such an interpretation in a pre-defined way may lead to a "biased" system. The last, but not the least, argument concerns maintainability. Since the knowledge is not structured and is only represented as a knowledge base, the implication of a change in a knowledge item cannot be anticipated nor assessed.

5.3.2 Extended Conventional Software Development

There are presently some theoretical studies which lead to consideration of another life cycle model and which tend to consider the KBS as an ordinary software production problem with its overall analysis completed prior to any implementation. KADS is the leader of this new way of design [5.1]. Many other examples may be found in Europe or in the U. S. [5.2, 5.3].

Based on a model-driven approach, these methodologies have much in common with conventional software development methodologies in that they prescribe phases, stages and activities, models, documents and deliverables. They provide specialized techniques, project metrics and quality assurance procedures for KBS development. They differ from other methodologies because of the essential differences between KBS's and more conventional systems that require considerably more complex models. It is important for these methodologies to present an approach at the analysis stage which is implementation-independent. For this reason, models are constructed at a high level of abstraction so as to emphasize the structure of the knowledge that underlies problem solving. The results-oriented approach of the development process lends itself to project management and quality assurance procedures.

For KBS development, these methodologies use a modified waterfall-type life cycle model to describe the KBS development process, based on the conventional systems development life cycle model of analysis, design and implementation. This is depicted in figure 5.2 with KBS dedicated aspects shown with dashed lines. These methodologies provide additional sets of supporting methods and techniques to account for areas such as expertise modeling and knowledge elicitation. They specify milestones, work packages and deliverables. Central to these approaches is the fact that thorough implementation-independent analysis and design are conducted before implementation, instead of the more commonly-used rapid prototyping approach described previously.

The fact that these methods are based upon conventional development technologies implies that they can be used in conjunction with other methodologies. They can even accommodate projects in which there is a shift in the solution, which means that the requirements documentation can be used as a starting point to continue the development in a conventional way.

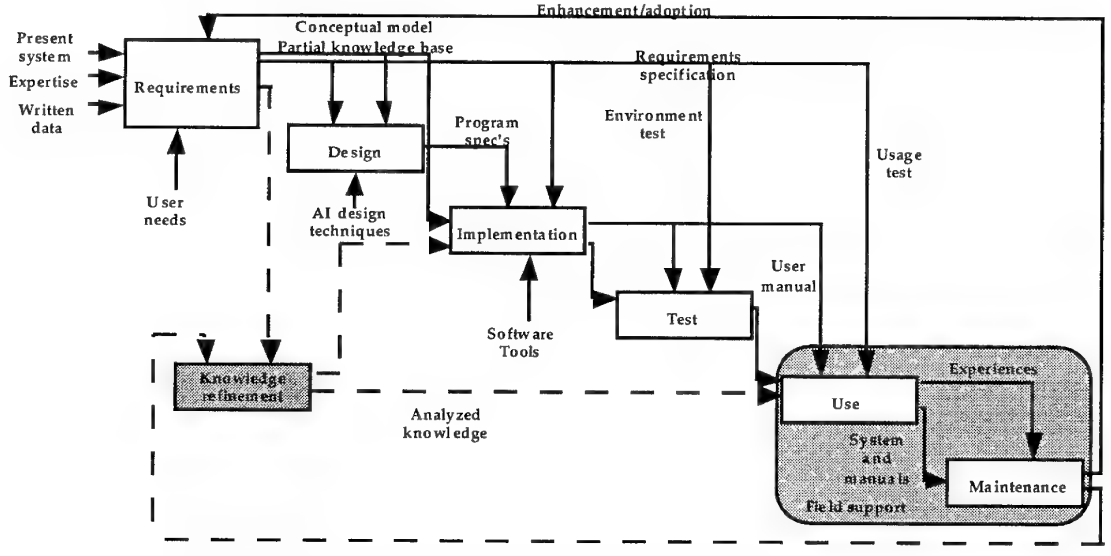


Figure 5.2 The KADS Top-Level Life-Cycle Model [5.1].

It is in the analysis and design phases that these methodologies differ most from conventional methodologies and therefore place the most emphasis. Analysis consists of external requirements analysis, broadly similar to conventional requirements engineering, and internal conceptual modeling, which has no parallel in conventional systems analysis. The latter involves the elicitation, interpretation, analysis and modeling of knowledge using an interpretation model to drive the entire process. In these methodologies, design is also model-driven, and there is frequently a direct mapping from the conceptual model of analysis to the functional and physical layers of the design model. Prototyping is viewed as a technique to assist a particular AI method or user interface. Prototyping is therefore an experimental or exploratory technique.

The extended conventional methods bring useful answers to the knowledge acquisition and maintenance issues mentioned in chapter 2.3.4.2 Current knowledge engineering practices heavily depend on interview techniques and the collection and analysis of notes. The process, although valuable, is slow and frequently paces the development activity. There is general agreement that considerable gains in speed and efficiency can be achieved by improving both tools and methods. Efforts aimed at the automation of portions of the knowledge acquisition interview process (as automatic learning or neural nets) have shown promising results. Likewise, on-going experiments in the direct collection of expert knowledge in performance environment simulations seem likely to result in tools that will dramatically speed and improve knowledge acquisition. At a different level, current initiatives are focused on the development time and expense for new systems in the future. All of these efforts reinforce the essential user driven knowledge engineering methodology and argue for its practical applicability to both KBS's and a wide range of conventional systems as well.

The nature of KBS's, and, in particular, their dependence on the elicitation of detailed and fine-grained knowledge from human experts, has motivated the development of particular systems engineering methodologies. This elicitation process usually results in a mass of unstructured and confusing paperwork. The task of sifting out elements of knowledge from this data is difficult. When a rapid prototyping methodology is used, this mass of seemingly unstructured data is often transformed into an equally unstructured mass of knowledge. This occurs mainly because the data obtained from the elicitation process and the knowledge in the implementation tool are at the same level of knowledge abstraction.

Several examples of existing methods are presented in the following paragraphs.

Knowledge modeling in KADS

In KADS, the process of building a KBS is seen as series of transformations of the verbal data until the implementable code is obtained as depicted in figure 5.3[5.4]. For knowledge acquisition, generic models may be selected from a library. These models can then be used to structure subsequent knowledge acquisition and analyses. This can be accomplished either directly by filling in the knowledge sources direct from text books and expertise transcripts, or indirectly by inferring the kind of knowledge that must be used to get from one meta-class to another. It is in this way that KADS eases the knowledge acquisition bottle-neck, rather than having any novel elicitation techniques.

A KADS four-layer model is a "paper model" of the expertise to be modeled and is very useful in order to capture expertise. It is not certain that all kinds of expertise can be reduced in such a way, but experiences tend to prove that the separation of knowledge into the four layers is more than simply a representational convenience and facilitate expertise use and keeping. It eases the understanding of the paper model both for the knowledge engineers and the experts (all the manipulated concepts are clearly defined, the goals of the KBS are organized, the treatments are represented as flow-diagrams understandable by the experts, ...). Such a conceptual model is the first output of the KBS development process and may be used for other developments.

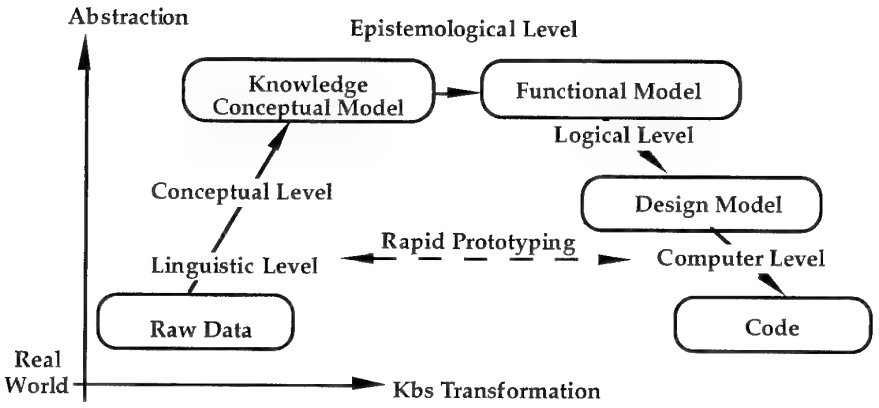


Figure 5.3 Level of Abstraction In KADS Knowledge Analysis [5.4]

KOD : Knowledge Oriented Design

KOD may be used for analyzing the expert interviews, and for collecting, validating, and structuring knowledge[5.5]. This method is mainly based on the two following ideas :

- structural analysis of objects manipulated by the expert and functions they are subject to;
- association of information to its statement conditions, so that knowledge is kept impregnated of the expert's point of view.

The proposed way of analyzing the expert talks is very complete and systematic and, subsequently, if it is strictly used, it may be extremely long and fastidious. KOD appears to be very efficient to model "classifying knowledge" or static expertise and the result is faithful to the expert talks. It offers an excellent bootstrap work when the expert system is started from scratch.

Dedicated Approaches And Related Projects

For specific applications, companies have developed either proper methodologies or methodologies derived from KADS and KOD (see f.i. [5.2]). Another example may be found in chapter 6.3.1.

Three ESPRIT projects are also involved within this field; KADS II (an advanced and comprehensive methodology for integrated KBS development), ACKNOWLEDGE (acquisition of knowledge) and VITAL (a methodology based workbench for KBS life cycle support).

5.3.3 A New Approach : Full Iterating Approach And Spiral Life Cycle Model

The main characteristic of the methodologies just described (thorough implementation- independent analysis and design prior to any implementation) may be viewed as an advantage or as a drawback. They do lack the dynamic link with experts that rapid prototyping offers. This characteristic could create huge differences between the expert and the paper model of his expertise leading to **a** time-consuming development cycle.

To cope with these limitations, an iterative KBS design approach, illustrated by the following spiral model (figure 5.4), was introduced [5.6]. Depending on the complexity of the system to be developed, such a life cycle model may be used as a stand-alone methodology, in the case of low complexity and small KBS's, or on top of a structured approach in the case of more complex or fuzzy domains. The prototype thus developed eases the functional analysis and allows early feedback from the end-user. This kind of model is very well suited to the knowledge acquisition and validation phases. The re-scaling of the developed system in its final environment needs to allow integration in conventional software and therefore, it may be more suitable to use a conventional software development methodology.

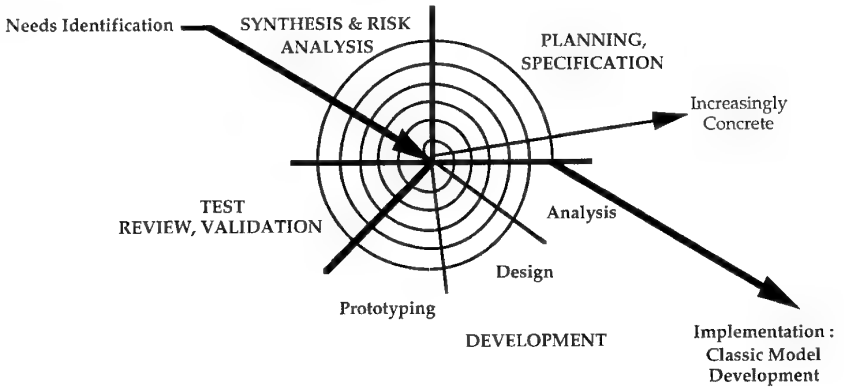


Figure 5.4 A Spiral Life-Cycle Model

Needs identification

This initial phase is intended to achieve an understanding of the domain and tasks of users, the organization and the environment . This understanding not only includes users' activities, but also the prevalent values and attitudes relative to productivity, technology, and change in general. Evaluative assessments of interest include identification of difficult and easy aspects of the task, barriers and potential avenues to improvement, and the relative leverage of the various stakeholders in the organization (experts, users, designers, ...). This problem framing can be viewed as a bootstrap phase preceding the spiral phases.

Synthesis and risk analysis

This phase is very important in the spiral approach. Its aim is to assess risk and identify feasibility based on results and progress to date. Areas of uncertainty which could constitute potential risks are also identified at this level.

Planning and specification

In this stage, plans, or updating of previously defined plans, are made for various development activities.

Validation, test and review

This phase can also be called quality assurance since the review of results of the activities of the development quadrant is performed. Scoping reports, minutes of meetings, review reports, notes, etc. are produced.

5.4 Tools for developing a KBS for G&C purposes

Available KBS development shells and "AI. languages" may not be suited to develop a KBS for G&C purposes. These tools were designed with the knowledge acquisition issue in mind, and not in order to optimize their real-time behavior and their resources needs. Moreover, they may not be used directly for safety critical applications. They are extremely useful and cost-efficient for developing off-line prototypes but, at the implementation level and for aeronautics systems, highly simplified architectures are required and the re-scaling of the prototype into its new environment has often to be considered. To ease this task, some constraints have to be put on development tools used in the previous stages :

- Capabilities of temporal reasoning and uncertainty management ;
- Tight communication with conventional software and handling software-hardware interrupts ;
- Facilities for focusing of attention on significant events and handling asynchronous inputs ;
- Use of fault tolerant hardware withstanding harsh environments ;
- Efficient memory use and management.

Even if some tools have been built explicitly for real-time applications [5.7], they are only a partial answer to the problems associated with real-time systems within this field.

Following the spiral life cycle model described in above, the need for infrastructure in each phase is surveyed and possible solutions are evoked. Software, hardware and integration aspects are covered along the different phases of the life cycle.

5.4.1 Development

It is here and in implementation that tool support is needed and fortunately, is available. This phase may be divided into analysis, design and prototyping.

5.4.1.1 Analysis Phase :

This phase is related to knowledge engineering. In connection with the methods described in the third part of this paper, tools exist or are under development. With such tools, the knowledge engineer is aided in fulfilling his task. His efficiency is improved mostly by allowing him to build links between manipulated concepts and in preventing him from *deviating* from the method. Some built-in editors are also useful in order to display hierarchy and graphs and to edit reports.

5.4.1.2 Design Phase :

To help to develop a multiple agent KBS, architecture simulators may be useful. Such a system has to interact with the outside and consists of several agents reasoning simultaneously and interacting asynchronously. It also has to take temporal constraints and limited resources into account. An architecture simulator helps the designer cope with the multiplicity of choices and complexity of interactions by helping him describe the architecture and evaluate its performance and, by doing so, improve it. Based on a description of the application (f.i. functional decomposition, structural description or use of resources), such a tool simulates the described architecture with chosen scenarios and supplies information on computer requirements such as memory, computer power, communication facilities , etc.

5.4.1.3 Prototyping phase :

Specific languages or expert system development tools may be used. Normally, the first choice is an overall knowledge representation framework, as described in chapter 4.2. Rule based, frame based, and logic based methodologies all have advantages and disadvantages. Systems may also be built in "traditional" programming languages (FORTRAN, C, ADA, etc.), specialized AI languages or, high level tools. The most popular specialized AI languages are PROLOG, LISP and SMALLTALK, along with their many derivatives. While they are reasonably general, they do provide a strong bias toward particular frameworks.

The high level KBS building tools attempt to insulate system builders from the details of programming. They, too, fall into several categories, based primarily on representation frameworks.

5.4.1.4 General development phase

During the development of a KBS, one has to couple it with the simulation facilities for testing. Used in severe environments, the KBS must be validated and completely tested. Prior to its operational use, it must be well integrated to its environment and communication facilities must be developed.

For particularly safety-critical applications, such as cockpit systems, it appears vital for the designer to have access to tools for verifying the correctness of the system. In fact, safety analysis replaces maintenance generally performed on traditional systems and which is forbidden for on-board ones. In real-time AI systems, the correctness of the system depends not only on the logical result of the system behavior, but also on the time at which the results are produced. The notion of verification of knowledge-based systems involves the checking of correctness, consistency, and completeness of reasoning performed. Most of the current literature that addresses this problem is strictly designed for checking the parameter values involved in the production rules. Although there is an increasing awareness of the special problems of real-time systems in the literature, relatively little work has been done on mechanizing safety analysis of hard-real-time systems, i.e. of those that are guaranteed to meet timing deadlines. Clearly, analyzing an AI real-time system is an even more difficult and challenging objective, but methods should be derived from techniques available for classic real-time systems [5.8].

In order to achieve a high interaction between the KBS and the user, the user interface must be extremely efficient. The goal is to reduce the time devoted to man/machine communication (the user must spend less time to understand the assisting system than to do the reasoning by himself ...). Since the communication between two humans is, most of the time, implicit, the KBS must display information such that the user sees what he wants to see (neither too detailed nor too short), when he wants and where he wants. These considerations induced the knowledge engineers to develop some "user cognitive models" for use when the task complexity is extreme (risk, uncertainty, time pressure, ...). Such a computer model serves to develop an intelligent assistance system which can

function as an automaton or as a support system. Such models have been successfully implemented in the defense area (within intelligent assistance system for fighter aircraft pilots), in the electrical industry (within nuclear power plant operators assisting systems) and in the steel industry (within blast furnace operators assisting systems).

5.4.2 Implementation

As the application is moved from the off-line to the on-line environment, new issues emerge :

- reasoning speed and time requirements satisfaction ;
- bottle-necks identification and treatment ;
- hardware architecture (communication, processor, memory, display, ...) ;
- man-machine interface ;
- machine-machine interface ;
- real-time performance evaluation ;
- ...

The last four points are directly related to the hardware. Generally, the improvement of real-time performance will be done at the expense of a loss of generality relative to the techniques used at knowledge acquisition level.

In order to cope with these issues, it seems relevant to consider simultaneously the three following aspects :

- the software - in order to guarantee its quality, to meet speed requirements and to satisfy memory constraints ;
- the hardware - in order to specify the CPU, memory and communications media ;
- the integration with the existing systems and with the user.

Software point of view

At the implementation level, the fundamental issue is no longer development cost but real-time behavior of applications ported in highly simplified architectures. The re-scaling of the prototype into its new environment, may lead to a complete rewriting of the off-line prototype in a more conventional language as C, C++ or ADA.

Hardware point of view

The hardware depends on many aspects and an error during its design may have drastic consequences. Even if it is impossible to forecast at the beginning of the KBS development what the hardware will look like, one has to take it into account as soon as possible. Tools and results, both practical and theoretical, exist (for example Petri nets) and it may useful to use them in order to reduce the risk of ultimate failure.

Real-time architectures are now widely available on the market. Such systems are more and more used for real-time implementation and a dedicated hardware is developed only for solving very specific functions for which real-time is harsh (e.g. mission planning).

Integration point of view

If the G&C intelligent system's architecture is considered as a distributed system consisting of a collection of autonomous data-processing knowledge sources controlled by a meta-planner, ADA seems to be a good substrata for developing such architectures. In these cases, knowledge sources and the meta-planner itself should be considered as potentially reusable software components.

If not, it is a case-by-case problem.

5.5 Conclusion

The successful realization of a KBS in G&C depends upon the careful consideration of all the specificities of such a system, such as performance issues, hardware constraints and limitations, etc. Tools are necessary for the development of such a system. The technology is available today to provide viable knowledge system solutions to well-chosen and well-defined problems. One can expect to see more and more successful on-board applications, as the research, the technology and engineering skills of application developers improve. It is presently an important and difficult decision to define the proper tools to populate a development infrastructure. Each project should be viewed as a specific case with no trivial generic answer in this matter.

This chapter gave preliminary clues for such a decision. However, improvements of infrastructure should be pursued in order to guarantee the success of KBS's in G&C. Among them, one can cite:

- the standardization of methodologies dedicated to KBS: all over the world, many companies are developing their own methodologies and nothing guarantees that it will be easy to switch from one to another ;
- the integration of these methodologies into the already developed software management tools or at least to consider the specificities of KBS ;
- the progress on verification and validation ;
- a better integration of AI tools in software development environments.

5.6 References

[5.1] Hickman, et al; "Analysis for knowledge based systems. A practical guide to the KADS methodology", Ellis Horwood Ed., 1989.

[5.2] Maesano Libero : "Modélisation conceptuelle des systèmes à base de connaissances", cours Human-Machine Interaction & Artificial Intelligence, Toulouse 1990.

[5.3] Ermine Jean-Louis : " Aspects méthodologiques et industriels des systèmes à base de connaissances", Journées état de l'art du CNRS, 1992.

[5.4] Dieng Rose: "Méthodes et outils d'acquisition des connaissances", ERGO-IA,Biarritz (Fr).

[5.5] Vogel Claude : " Génie Cognitif", Ed. Masson, 1988.

[5.6] Boehm, B. "A spiral model of software development and enhancement", IEEE Computer, pp 61-72, May 1988.

[5.7] Sallé Stéphane E. & Årzén Karl-Erik : "A comparison between three development tools for real-time expert systems : Chronos, Muse and G2", CACSD, Tampa (Fl - USA), 12/89.

[5.8] Grisoni M. & Howells H., European workshop on V&V for KBS, CAmbridge UK, 22-24 1991.

[5.9] Amalberti René & Deblon François : "Cognitive modeling of fighter aircraft process control : a step towards an intelligent onboard assistance system", Int. J. of Man-machine studies.

[5.10] Rouse William : "Design for success : A human centered approach to designing successful products and systems", Wiley series in systems engineering, 1991

CHAPTER 6

EXAMPLE APPLICATIONS

6.1 Aeronautics Applications

6.1.1 CASSY: A Cockpit Assistant System for Instrument Flight Operation

6.1.1.1 Introduction

In this section, an application example for a knowledge-based cockpit assistant system is described which was developed to support the pilot crew of commercial aircraft during IFR (Instrument Flight Rules) flight operation. This application corresponds to the domain of the guidance and control aircraft level, as described in section 2.2.1. Commercial air transport is characterized by flights under IFR, since this kind of operation provides a high degree of independence from adverse weather conditions. However, lack of visual references of the aircraft environment and increased complexity of automated cockpit instrumentation can result in overloading the pilot crew. It is a fact that by far the majority of accidents are caused by some kind of human error or mismatching [6.1, 6.2]. Aircraft accident causes can be correlated with findings on error prone characteristic features or shortages of human cognitive behavior [6.3 through 6.9]. From these findings, it becomes evident that knowledge-based electronic pilot assistance can lead to effective support for pilot tasks like situation assessment, planning and decision making, and plan execution. This requires a situation-specific design philosophy for complementing human capabilities by autonomous machine capabilities. In this sense, it is not just an allocation of functions for automation (replacing human control functions by automatic ones) and relying on human adaptability for what is not covered by automation. It rather means that the piloting sub functions - except the control actions themselves - are performed in a cooperative manner more or less in parallel by the pilot crew and the cockpit assistant system.

On the basis of formal knowledge about the situation-specific user needs, a cockpit assistant for IFR operation has been developed. This research and development work led to a first prototype at the Universität der Bundeswehr München, called ASPIO (Assistant for Single Pilot IFR Operation) [6.7, 6.10]. Flight simulator trials with ASPIO have shown that this approach proves very effective. Since 1991, development of an advanced Cockpit Assistant System (CASSY) [6.11, 6.12] for the two man crew has been underway in cooperation with Dornier Luftfahrt GmbH and has been implemented on SGI workstations in conjunction with the flight simulator facilities of both organizations.

6.1.1.2 Functional Structure of CASSY Modules

Similar to the basic structure of knowledge-based assistant systems described in chapter 2, CASSY comprises the following task specific modules as shown in Figure 6.1:

- Situation Assessment Modules as Monitoring of Flight Status(MFS), Monitoring of Systems (MS), and Monitoring of Environment (ME),as well as Piloting Expert (PE) and Pilot Intent and Error Recognition (PIER) for behavioral interpretation
- Automatic Flight Planner (AFP)
- Piloting Expert (PE)
- Pilot Intent and Error Recognition (PIER)
- Dialogue Manager (DM) as coordination module
- Execution Aid (EA)

The internal coordination function, as described in section 2.1, is not treated as a separate module in CASSY because of communication efficiency. This function is embedded in the situation assessment modules and the AFP.

As long as there is no data link capability available, the Air Traffic Control (ATC) interface is not explicitly realized within the cockpit assistant. The important ATC instructions and information have to be transferred to CASSY by the pilot crew through speech input. This can be operationalized rather easily because the ATC instructions are to be acknowledged verbally by the pilot, anyway.

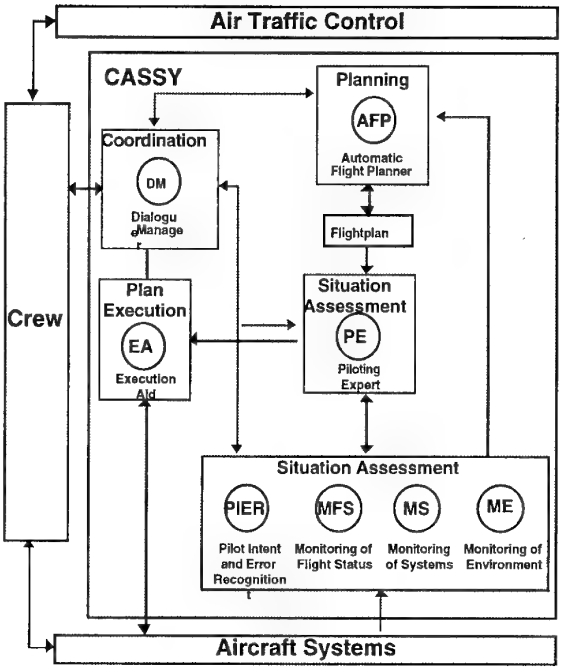


Figure 6.1 Functional Structure of CASSY

In the following sections, the CASSY modules are described in more detail, focusing on those modules where knowledge about the pilot crew concerns most.

6.1.1.2.1 Modules for Situation Assessment

Situation assessment is achieved by a situation representation on the basis of the flight chronology and potential tasks and events pertinent to the flight segments. A data base contains the actual values of state variables to be instantiated in that representation (see section 2.2.1). Situation evaluation as part of the situation assessment is accomplished by means of a number of monitoring modules along with the Pilot Intent and Error Recognition module (PIER) and the Piloting Expert module (PE):

The Monitor of Flight Status (MFS) has the task of monitoring the flight progress in order to be aware of the arrival at any subgoal of interest during the flight. On the basis of this information, the actual state with respect to the flight plan can be actualized and the expected pilot actions can be generated in the PE module. Also, standard callouts, usually delivered by the co-pilot in the conventional two man cockpit, can be provided via messages to the speech system in the DM.

The health status of the aircraft systems is monitored by the Monitor of Systems (MS). This module works on the information about defects in the aircraft systems in order to generate messages to the pilot crew. This information is also rendered to the AFP and the PE, since replanning could be necessary and the expected pilot actions could be different subsequent to system defects or failures.

Sensor information for the assessment and evaluation of weather conditions and surrounding traffic are gathered in the Monitor of Environment (ME). This monitoring module also identifies deviations from normal with respect to the immediate aircraft environment and reports to the pilot crew and the CASSY modules concerned.

On the basis of the flight plan as generated by the AFP (see section 6.1.1.2.2) and acknowledged by the crew, the PE is capable of determining the expected plan execution actions the pilot crew is supposed to carry out during the various flight phases. So far, this model describes the pilot crew rule-based behavior as a normative reference [6.15, 6.20]. In a later development state, it will also include a model of the individual situation-specific normal behavior

of the pilots actually flying. The model mainly consists of ranges of expected pilot actions. However, it also represents the knowledge about the danger boundaries for pilot actions, which is of fundamental importance for the function of monitoring of pilot behavior within the PIER.

The rule-based normative model comprises all flight phases and mainly the following flight task domains:

- Primary flight guidance (altitude, course, speed)
- Control of landing flaps, landing gear, spoilers
- Radio navigation procedures
- Procedures for take-off, holding and approach
- Check list procedures
- Communication with air traffic control

Typical properties of the pilot crew tasks with regard to temporal and causal aspects (i. e., rule-based execution, parallel execution of different tasks, different situation-dependent rule packages for the same, elementary flight task (such as radio navigation)), task breakdown in "flight phase-related" and "event-related", and typically awaiting certain events for the execution of a short action, led to the conclusion that petri nets are to be preferably used for the representation [6.15]. The net structure is hierarchical and modularized. Subnets might work in parallel or alternatively.

The PIER function bears upon not only the information of situation assessment and monitoring but also on reference information about the pilot behavior as well as tolerances with regard to safety margins from the PE.

The main task of the PIER consists of the recognition of either pilot intent or error in order to explain unusual discrepancies in pilot behavior and flight status from what should be expected according to the flight plan [6.16]. The intent recognition is initiated upon detection of pilot actions deviating from the flight plan. In case of a confirmation of a change of the pilot intent, the aim is to assess the intent or the new flight plan the pilot has adopted. These cases of off-nominal unannounced change of pilot intent are likely to occur in the context of IFR flight operations when any of the following kinds of situations are encountered:

- Flight in a conflict area (for example thunderstorm, turbulence, collision hazard, etc.)
- Flight around a conflict area and return to the original flight plan
- Selection of a new waypoint and no intent to return to the original flight plan
- Reaction due to a system failure event
- Abortion of take-off or final approach

The intent recognition is mainly performed by use of an inference algorithm based on known intent hypotheses (see figure 6.2)[6.16] . A priori probabilities are assigned to the possible hypotheses with regard to what is known about the actual situation. On the basis of the actual pilot actions, these probabilities are modified. The most probable hypothesis is the best candidate for selection. With this kind of approach either pilot crew intent or error is inferenced.

The crew behavior is classified as an error only if a meaningful intention was determined or if the danger boundary is going to be exceeded. In these cases warning messages are transferred to the crew through the DM module.

If intentional behavior has become evident but the intention is not completely uncovered a short hint is given to the crew. Possibly, the pilot crew might react to the hint by speaking out new directives. In any case, the module carries on trying to specify the crew intent by further checking certainty factors of the possible intent hypotheses.

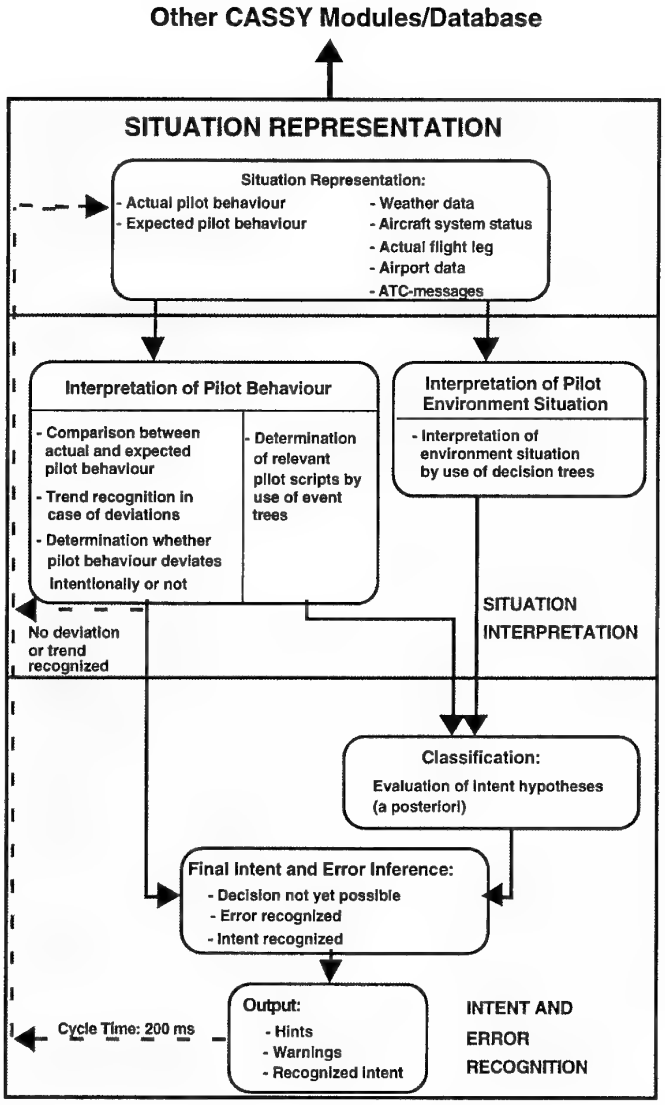


Figure 6.2 Functional Structure of the Pilot Intent and Error Recognizer (PIER) Module

6.1.1.2.2 Automatic Flight Planner (AFP)

During flight, the AFP autonomously generates recommendations for modifications of the flight plan when significant deviations from the current flight plan have to be faced because of events such as new ATC instructions, adverse weather conditions, system failures or insistently deviating pilot actions [6.13]. The pilot crew represents the decision issuer with regard to acceptance or rejection of the AFP recommendations. The crew decides about the actual flight plan by means of communication links with the AFP through the DM. They essentially decide on a globally but unambiguously defined plan about the destination, the route, necessary flight procedures and the vertical profile. On the basis of this decision, optimizing routines are providing the exact flight plan data as the basis for the forthcoming flight operation. At this point it should be clarified that new flight plan recommendations are not only produced autonomously as a result of autonomous situation assessment but also on explicit request of the pilot crew. Flight plan changes might be also initiated by information from the PIER concerning recognized changes of pilot intentions not explicitly fed into the system by the pilot crew [6.19].

6.1.1.2.3 Dialogue Manager (DM)

The Dialogue Manger (DM) is part of the coordination function, as described in chapter 2. It comprises all components for the information transfer between CASSY and the pilot crew, including the management of information flow to the pilot crew.

Extensive use is made of speech communication in either direction between pilot crew and CASSY [6.17, 6.19]. Speech input is used for acknowledgment and rejection of assistant system recommendations and as a medium for instructions to the EA. The latter can be used optionally by the pilot crew instead of conventional pilot controls and inputs for aircraft systems other than CASSY. Speech input is mechanized by means of a speech recognition system. The vocabulary is based on the standard phraseology of civil aviation. Specific algorithms are developed in order to provide support for the speech recognition function through context knowledge. Hereby, the DM controls and switches syntax depending on the flight situation.

Synthetic speech is used for speech output, with different voices for different categories of assistant messages. More complex information, like comprehensive flight plan recommendations, is presented through the visual display of CASSY.

A priority control function for the message flow to the pilot crew is one of the core DM functions. It consists of the priority assessment (message ranking), the dialogue control and some buffer function, as depicted in Figure 6.3. This function decides which incoming information from the CASSY modules will be presented to the pilot crew as relevant messages, and in which order (time) it will be presented. The ordering is executed on the basis of the situation interpretation, certain general rules and the time when the message is coming in. The message transfer list is buffered until the next message cycle. The information presentation to the pilot crew is mechanized by both speech output and an graphic/alphanumeric display.

The communication between DM and CASSY interfaces, especially the pilot interface components, is organized by inter-process communication and standard interfaces for data transfer. In general, the inter-process communication for both inside the CASSY modules and between CASSY modules (including interfaces) is organized by both message buffer transfer and transfer via shared memory. The shared memory contains all data of possible common interest for one or more CASSY modules, except NAV data. There is a special service process for NAV data, which delivers a NAV data subset from a given data base (disc) as explicitly requested from one of the other CASSY processes.

6.1.1.2.4 Execution Aid (EA)

In order to support the pilot crew in the course of executing the actual flight plan, the EA offers a variety of optional service functions, conventionally performed by the co-pilot without automatic aiding. Among these functions are automatic instrument setting, flap and gear setting and navigational calculations. All of these functions can be commanded by the pilot crew through speech inputs.

6.1.1.3 Experimental Test Results

The cockpit assistant system implementation in a flight simulator facility at the Universität der Bundeswehr München is continuously extended. A one-seat fixed base cockpit with computer generated outside vision and head down display, artificial stick force as well as speech input and output is employed. The simulation comprises aircraft dynamics (6-degree of freedom model), autopilot system, radio navigation systems and a wind model.

A first prototype implementation of the cockpit assistant system was tested late 1989. This implementation, named ASPIO, differed from CASSY in a number of ways.

ASPIO was intentionally limited to support a single pilot and was not able to consider and recognize widely deviating pilot intentions, characterized by the use of an isolated-word speech recognizer, and was implemented only for the flight phases from arrival through final approach. This prototype comprised the main functions, though, of a knowledge-based cockpit assistant system, which are necessary for a serious flight simulator evaluation.

The experiments were aimed at proving hypothesized enhancements in overall performance and safety without increase of pilot workload. For this purpose, performance criteria like flight accuracy, pilot errors in situation assessment and system operation, duration and quality of planning, pilot workload and pilot acceptance were investigated.

The pertinent parameters were evaluated for a number of test runs of IFR approaches. Three different IFR scenarios were developed which consisted of standard situations as well as of unusual events and emergencies. A total of nine professional pilots, each with a great amount of IFR flight experience, performed these test runs. Some of the evaluation results [6.7, 6.10] showing flight accuracy and duration of planning are presented in the following.

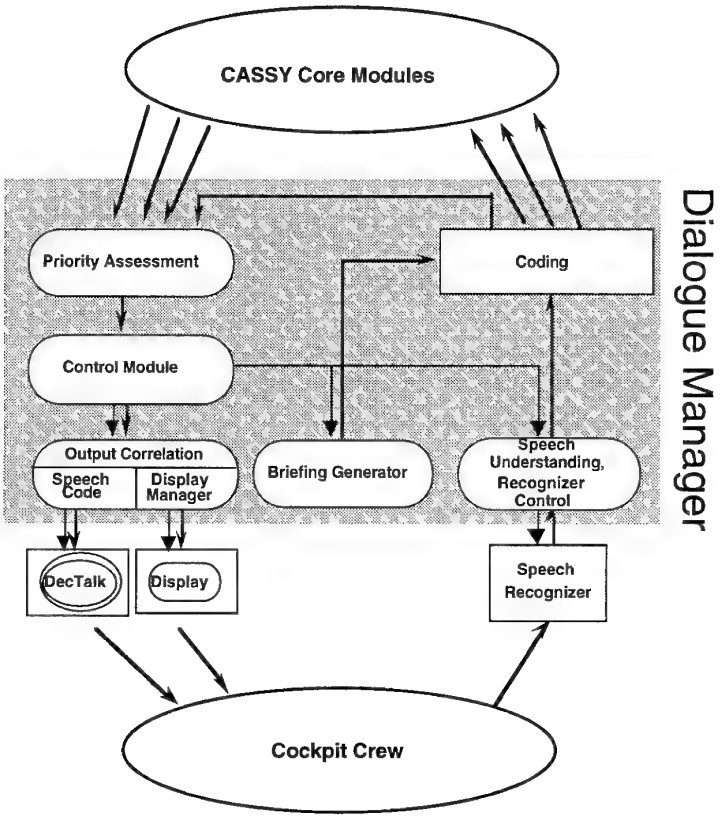


Figure 6.3 Functional Structure of the Dialogue Manager (DM) Module

As an indication of flight accuracy, the airspeed deviation was considered to be most appropriate for evaluation. It was supposed to be controlled manually by the pilots. The evaluation results for the standard deviation of airspeed for all pilots are depicted in Figure 6.4. It was shown that the improvement in flight accuracy by use of the assistant functions was highly significant.

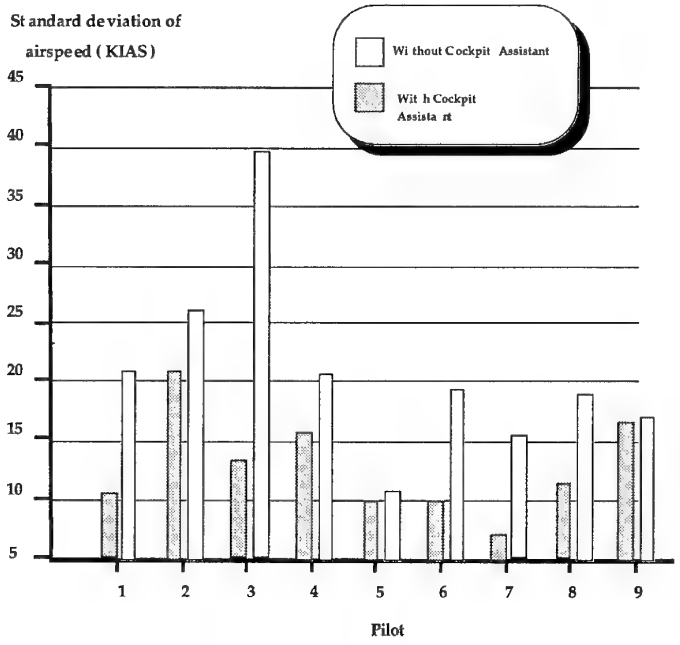


Figure 6.4 Flight Simulator Results with ASPIO: Performance in Flight Accuracy

In figure 6.5, the pilot's decision time without support by the cockpit assistant is shown in comparison to that with assistant aid. The planning activity was triggered by an ATC instruction, where an immediate reply about the pilot's further intentions was demanded. The time between the ATC instruction and the pilot's reply was measured.

For the particular example, shown in figure 6.5, the planning and decision task was to check for an alternative approach procedure taking into consideration the weather and airport data, after information was given that the instrument landing system at the destination airport had failed. The differences are obvious. Moreover, the pilots stated that all decisions recommended by the cockpit assistant, made sense.

With regard to pilot errors, no unusual deviations from normal flight could be observed when the pilot crew was supported by the cockpit assistance function, although a number of pilot errors occurred during the simulated test flights. The experimental data showed a tendency of slight reduction in pilot workload with the cockpit assistance function activated and the subjective scores about the cockpit functions as a whole were positive.

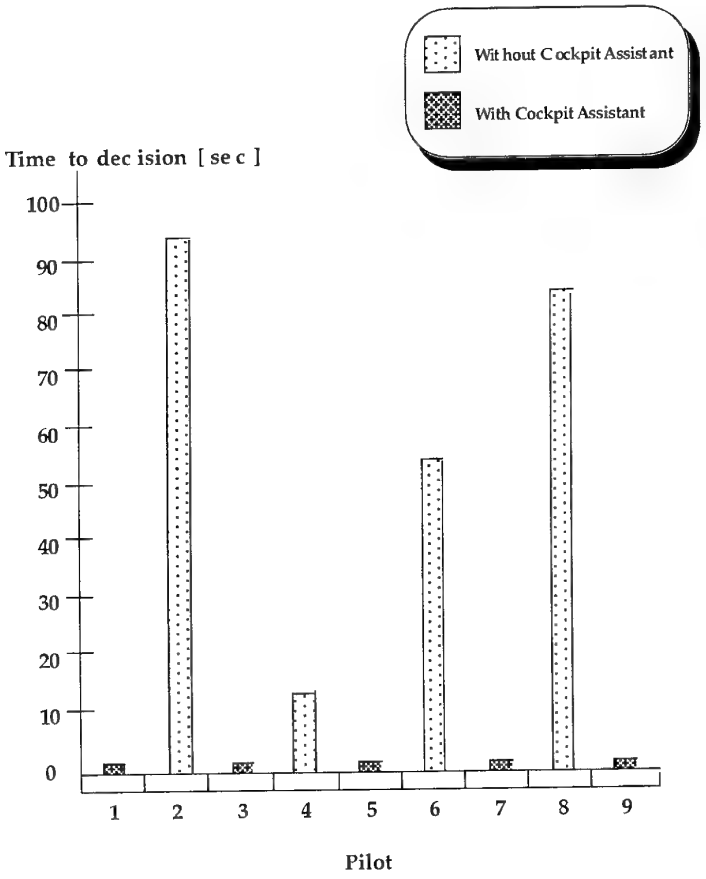


Figure 6.5 Flight Simulator Results with ASPIO: Performance in Planning Tasks

6.1.1.4 Concluding Remarks

Whenever the piloting crew tries to act sensibly in the face of a flight situation, they act predictably within a confined range of possible actions. This hypothesis has not been verified yet, in its full sense, because of the difficulties of representing the actual situation in flight with all its implications and contingencies - that is the state of the world including that of the pilots themselves, i.e. their error susceptibility, too. However, this fundamental hypothesis is the basis for the potential success of knowledge-based cockpit assistant systems for supporting the pilots in tasks like situation assessment and decision making.

Therefore, understanding of the state of the world, i.e. the flight situation, including the potential pilot crew reactions, is the basic design requirement for an assistant system. This is so because pilots can not be expected to convey their own perceptions and intents to the assistant system.

CASSY is able to understand the flight situation on the basis of knowledge about facts and procedures of the piloting task and aviation domain and the actual data about the flight status, pilot actions and intent. The situation is evaluated with regard to conflicts concerning the actual flight plan. If necessary, the system derives a revised flight plan as a recommendation to the pilot or can serve the pilot for plan execution tasks. It also issues advisories or warning messages to the pilot crew in case of recognized pilot errors.

The assistant system performance has been investigated in a first flight simulator study, based on an implementation of a prototype system, called ASPIO. The results were extremely promising. CASSY will undergo a similar investigation and will be tested in flight in early 1994.

6.1.2 The Pilot's Associate Program

The DARPA - USAF Pilot's Associate Program has developed a prototype knowledge-based Decision Support System for pilots of single seat fighter aircraft. The current prototype is running in a realistic concept cockpit for an advanced tactical fighter aircraft. It operates in a full mission flight simulation system that projects a realistic, demanding 1997 air combat scenario. The system runs in real-time, and is capable of processing over 100 "enemy" objects per second using avionics boards in a VME chassis, similar to those found on current aircraft. The existing Pilot's Associate is considered to be a realistic prototype of an avionics decision support system that will be able to support the operational needs of future tactical pilots.

As an Associate system, it provides the pilot a variety of services similar to the functions performed by the Weapons Systems Operator in current two-place tactical aircraft. The associate nature of the system is, in fact, derived from a model of the interaction between two crew members in such aircraft. Thus the system is active, in the sense that it attempts to anticipate the needs of the pilot to provide answers at the time and in the form that are most useful to the pilot [6.21-6.24]. It does not, however, attempt to dictate solutions to the pilot.

Although the pilot can express his preferences and mandate action by the system, his direct input is generally not required for most system functions. Thus the system collects it operational data directly from the aircraft sensors, forms its own problem definitions, develops solutions and, in certain special cases - authorized by the pilot prior to mission execution - carries out the plans with little explicit direction from the pilot. In every case, however, the plans must conform to the pilot's own plan, and the pilot must be able to recognize that fact. If not, he can override or reject any recommendation either explicitly, or by merely pursuing his own, different plan. This action will then be recognized by the Pilot's Associate system and it will adjust its plans accordingly.

The Pilot's Associate serves as a realistic example of the potential for application of knowledge based technology to current and projected operational problems. It has demonstrated the ability of such systems to support humans in forming valid models of real world conditions (situation awareness), developing effective plans for the solution of difficult problems (response planning) and responding to time critical situations (emergency response) under the stress of combat conditions. Although the Pilot's Associate was never intended for full scale development, it has motivated follow-on programs as well as system requirements for aircraft in development. It seems very likely that the heritage of this important effort will be visible in fielded systems in the early part of the next century.

6.1.2.1 Program History

The Pilot's Associate Program was launched in February 1986 as an application demonstration in DARPA's Strategic Computing Program. The technical goal of the Program was to encourage development of real-time, cooperating knowledge-based systems and speech understanding. The U.S. Air Force initial interest was in exploring the potential application and benefit- expressed as survivability and mission effectiveness - of that technology in future military aircraft. Specifically, the Program was directed toward an on-board system in an advanced single-seat fighter of fighter-bomber aircraft.

As shown in Figure 6.6, the Program was planned as a five-year, two phase effort, administered for DARPA by the USAF Wright Aeronautical Labs. Phase One focused on a proof of the concept of cooperating expert systems and indications of potential benefit to aircrews. In the initial phase, two major Demonstrations of increasingly capable, but functionally limited, non-real-time Pilot's Associate System were held. Two independent development teams led by Lockheed Aeronautical Systems Company and McDonnell Aircraft Company implemented separate Pilot's Associate Systems applied to distinct combat domains (air-air and air-ground). The Phase Two effort was dedicated to air-air issues and conducted entirely by the Lockheed team.

In Phase one, successful demonstrations in March of 1988 (Demo 2) and November 1989 (Demo 3) illustrated the viability of cooperating expert systems in a supporting complex, interactive processes of the fighter aviation

domain. This, coupled with the observed potential for significant benefit - improved effectiveness and survivability - motivated a decision to proceed with a second Phase of the Program [6.25 - 6.27].

Phase Two focused on issues of practicality and application to actual aircraft. Thus, a principal goal of this phase was to develop a prototype system that would, first, run in "real-time" and secondly, operate on hardware that was similar to that expected to be seen in the emerging generation of aircraft. In addition, experiments in improved knowledge acquisition techniques and in verification and testing concepts were planned.

Work began in late 1989 and was completed in the spring of 1992. After six weeks of testing with operational US Air Force pilots, a final demonstration (Demo 4) was held in July 1992. The test and the ensuing demonstration featured a demanding air-air escort scenario in which pilots were to precede and provide protection for a strike package of vulnerable ground-attack aircraft penetrating a modern integrated air defense system. Although the testing results are still in final analysis, the collected impressions of the test pilot cadre have been favorable and indicate satisfaction with the system.

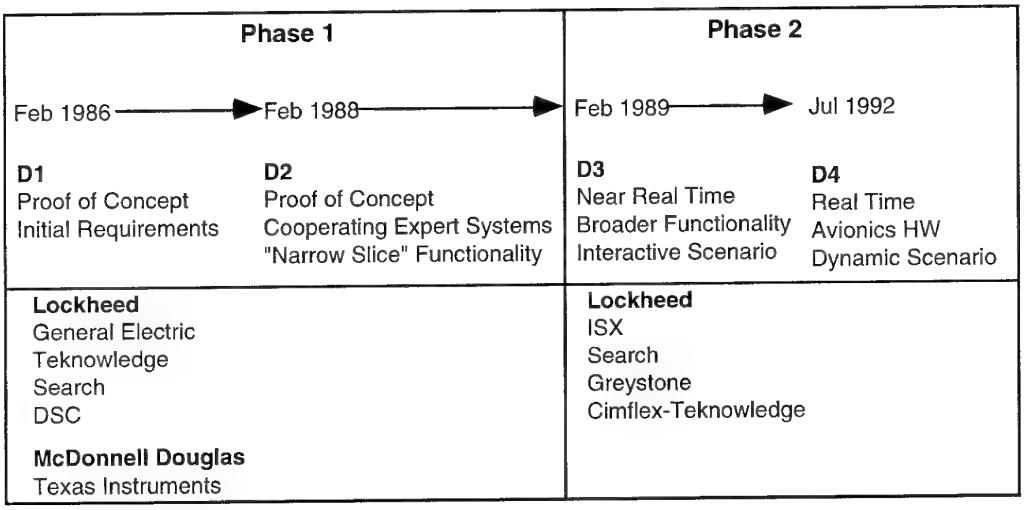


Figure 6.6 PA Development Schedule

6.1.2.2 Pilot's Associate System Architecture and Functionality

The Pilot's Associate was originally designed around an architecture that reflects the high-level functions it supports. The PA architecture is shown in Figure 6.7 Individual modules, which were originally distinct expert systems, were developed and integrated. Two modules, known as Systems Status and Situation Assessment, essentially correspond to the Monitor and Diagnosis functions of the functional model described in Chapter 2 of this report.

System Status (SS) monitors and analyzes onboard systems data to determine current aircraft state and evaluate actual systems capabilities. Malfunctions are evaluated in current mission context to determine the degree of effective degradation and impact on current and proposed tactical and mission plans. When possible, the system also generates remedial plans and coordinates them with plan generation in other systems. Thus, this sub-system, as do all other modules, subsumes small portions of Plan Generation and Coordination of the general functional model discussed in Chapter 2.

The Situation Assessment (SA) module monitors events external to the aircraft to provide a combat situation analysis. It combines stored mission data with sensed data from on-board sensors and cooperative sources to provide context sensitive information directly to the pilot and mission state information to other modules. Thus, it combines the requirement to react to and interpret unexpected events with the query-driven function of interrogating the external space with respect to data needs of current or proposed plans.

The Pilot Vehicle Interface (PVI) module partially supports the Coordination function in the general functional model, as well as providing elements of the Diagnosis and Planning function. The PVI is responsible for explicit, and importantly, implicit communication with the pilot and interface to the cockpit display and command systems. The design integrates interface management, an adaptive aider, and an error monitor to accomplish monitoring, diagnosis and planning functions.

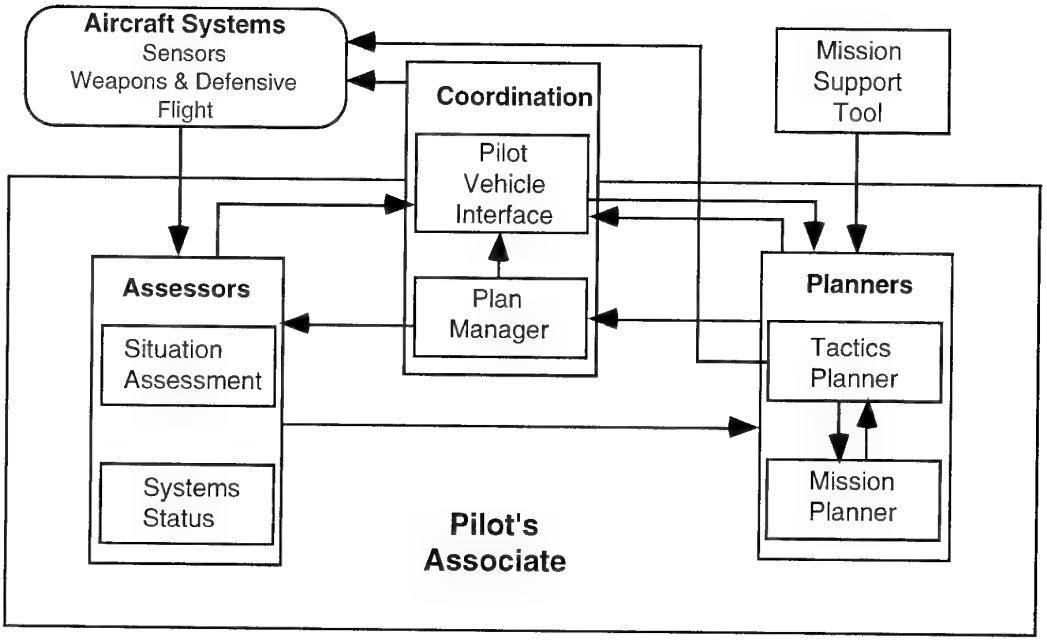


Figure 6.7 PA Architecture

Pilot intent determination is the essential function of the PVI. Through monitoring pilot actions and mission context and comparing them to a model of the pilots plans and goals for the mission, the system determines pilot intent without the need for explicit input. These intentions may be passed to the rest of the system as implicit commands. Although the pilot can always request explanation and override recommendations, in most cases, the pilot is informed by observation of system behavior - much as he is by actions of another crew member. Actions that are consistent with recommendations are inferred to be approval. Deviations from any current plan indicate a rejection of that plan and a preference for another action. Whenever the system cannot infer pilot intent it responds with standard default behavior, similar to current avionics systems.

Plan Generation and Plan Selection are supported by the Tactics Planner (TP) for tactical plans and the Mission Planner (MP) for route planning. The Mission Planner creates and maintains a takeoff-to-landing mission profile that includes routes, resources, and time constraints. Beginning with a start and end point, it generates a three dimensional flight profile that trades exposure to known threats (ground-based and airborne) with consumption of resources (fuel, expendables, etc.). The resulting 4-D route, including projected exposure to threats and consumption budgets, are provided to other systems to other modules to reason about and, in the case of PVI, for display to the pilot. MP also functions as a resource for the Tactics Planner, providing minimum risk trajectories to support both offensive and defensive plans.

The Tactics Planner reasons about threats and targets and plans for sensor actions, attack , or if required, evasion. TP also tasks the SA module to monitor certain high-value objects. The TP does not "invent" new tactics. Rather, it recommends tactics, selected from the pilot's own set of pre-briefed tactics, or library of stored "standard" tactics, that are most suitable for the actual situation. In doing this, the TP applies expert heuristics to an array of sensory data to find very good solutions to tactical problems more rapidly and in finer detail than humans can. The current TP design will be described in greater detail at a later point.

Except in certain select cases - authorized defensive expenditures and emissions, for example - Plan Execution is largely accomplished by the pilot. Authorization is provided by the pilot through a sixth module, the Mission Support Tool (MST). The MST functions as a ground based Mission Planning System. The pilot uses it to plan the mission, select sensor actions and search patterns for various conditions, and to express a preference for both offensive and defensive tactics in expected situations. This data is then transferred to the Pilot's Associate through magnetic media and used to initialize the Pilot's Associate and customize it for individual pilot techniques and preferences for the particular mission.

The Pilot's Associate interacts with the pilot through display and control systems in the cockpit, Figure 6.8. These include five flat panel displays, a multi-function "Hands-On Throttle and Stick System" (HOTAS). The displays include an aircraft Systems Display that graphically depicts the state of various systems in several formats, ranging from traditional "round dials" to schematics. This display also indicates the quantity and state of on-board weapons. A Weapons and Sensor Display System indicates the state and field of view of air-air missiles as well as that of the radar and IR sensors. The primary displays are three tactical displays that are somewhat interchangeable at the pilot's discretion. A Tactical Situation Display provides a plan view of the tactical situation, including map information, route, ground-based threat location, and aircraft locations and state data. The Offensive Situation Display provides offensive tactics information, and in particular, targeting and weapons information. The Defensive Situation Display provides greater detail on identified threats and suggested reactions to the threats. Important, time critical information, such as missile warning and authorized PA response (e.g. using "chaff" or "flares"), are relayed through a voice interface system. That same system also allows the pilot to verbally control displays and certain other functions. In addition, a small message screen provides for text-based communication with the pilot.

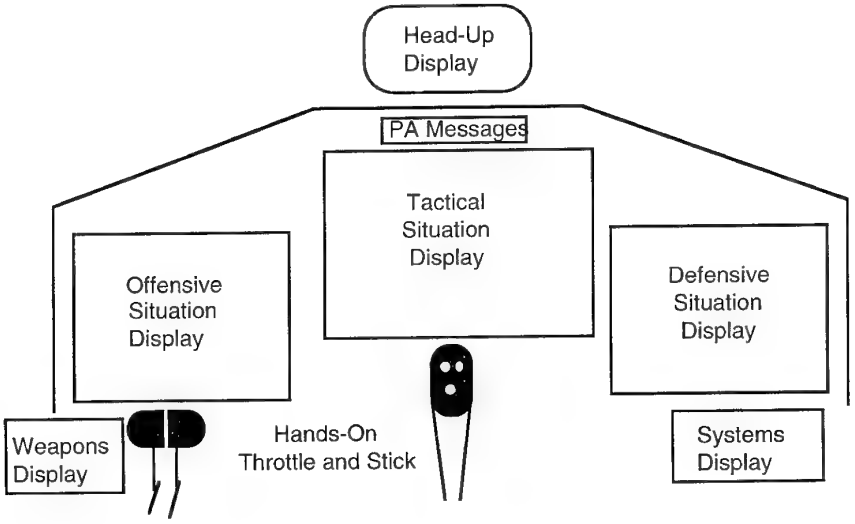


Figure 6.8 PA Cockpit

The Phase One prototype was a heterogeneous, loosely coupled system in which independent modules, using domain specific approaches, communicated through a global blackboard. In Phase Two, the perceived exigencies of real time processing drove the design to a much more tightly-coupled implementation.

The system has been designed to run on five 68040 boards, mounted in a VME chassis. Each board contains 16mb of memory, with most of the memory on each board available to any processor on any board. The hardware also includes an additional 2mb of separate global memory. The operating system is VxWorks, which is a variation of UNIX specialized for real-time performance in a distributed processing environment. A VxWorks process is executed only when signaled by a global event flag that can be observed by any processor in the system. That signal then unlocks the process and allows its execution. Thus, there is no processing overhead with unsignaled, or blocked processes.

In the Phase Two design, the collection of five independent expert systems, or modules, evolved into six closely related sub-systems specializing in the functions previously described. The new, sixth sub-system, dubbed Plan Manager, serves to facilitate communication among the other five subsystems. Despite functional differences, all

sub-systems share a common view of the domain, seen in a representational structure called the Plan-Goal Graph (PGG). Sub-systems develop plans that are represented as nodes in the PGG. When a plan is built, it instantiates the appropriate node in the Plan Manager's plans database, which is then available to other systems that support or are supported by the plan.

6.1.2.3 Tactics Planner Design

Of the 6 sub-systems that comprise the Pilot's Associate, perhaps the best example of the use and evolution of knowledge based technology is seen in the Tactics Planner. From one perspective, this results from the somewhat amorphous and context dependent nature of the tactics planning problem operational environment. Tactical problems tend to combine highly-valued, complex choices with considerable urgency. In general, these needs are met with adequate, if not optimum, "satisficing " solutions as discussed in Chapters 3 and 4. Expert heuristics, captured in knowledge-bases offer a technical approach to support for such problems.

In Phase 1, this system was implemented in Lisp using the KADET development tool planning structure. At the time, when the program focus was on rapid functional development and test, this tool met all needs. However, the demands of real-time planning mandated a shift to a design in which execution speed could be studied and optimized.

In the Phase Two Pilot's Associate, the Tactics Planner is implemented as a generative planner, employing knowledge-based planning techniques described in Chapter 4 of this report. When signaled, the planner selects stored skeletal plans determined to be appropriate to the actual conditions and then specializes their attributes to provide an effective response. Figure 6.9 illustrates the general sequence of events in the generation of such a plan.

The process is initiated when the Situation Assessment (or Systems Status) sub-system detects and identifies an "interesting event" that calls for a tactical response. This is effected by SA and SS structures known as *monitors.* Monitors are set to watch for particular occurrences, such as the initial observation of an aircraft, a change in the assessed threat value of some known aircraft or the launch of an enemy missile. Any such event then results in the firing of the corresponding monitor and the posting of an event signal. The Tactics Planner then selects goals appropriate for the situation and develops a strategy for satisfying those goals. Resources are then allocated and the plan is executed. During execution, the selected plan is continually modified to adapt to changing conditions until the plan ultimately completes or fails.

Planning data is received from three sources. Prior to the mission, internal data structures are initialized with specific data particular to the mission and preferences of the individual pilot flying the mission through the Mission Support Tool. In this manner, the plans contained in the plan database are modified to reflect the requirements of the mission and the tactical decisions of the pilot. During the mission, the Tactics Planner continually reviews updated data regarding aircraft and mission status in the Mission Systems and Threat databases.

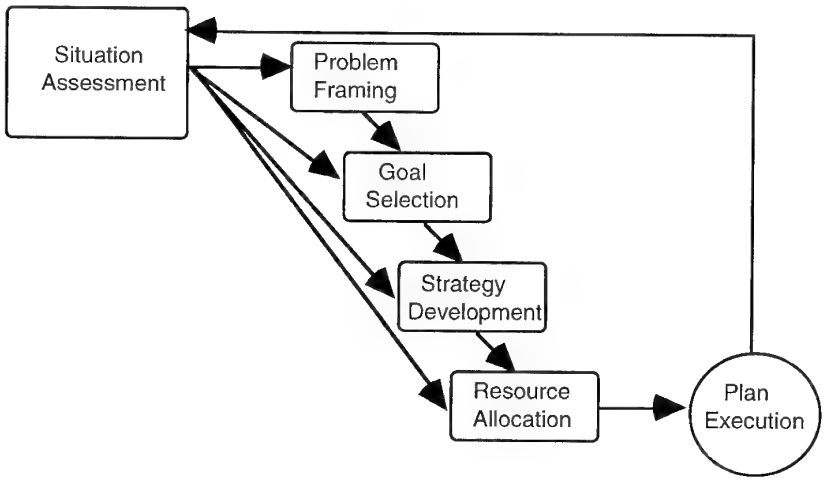


Figure 6.9 Simplified View of Plan Generation in the Tactics Planner

Within the Tactics Planner, a five level data structure known as a "Focus of Attention (FOA) hierarchy" provides planning control and knowledge organization. As illustrated in Figure 6.10, this structure maps directly onto the Plan and Goal Graph knowledge representation, with each level in the hierarchy corresponding to a node in the PGG.

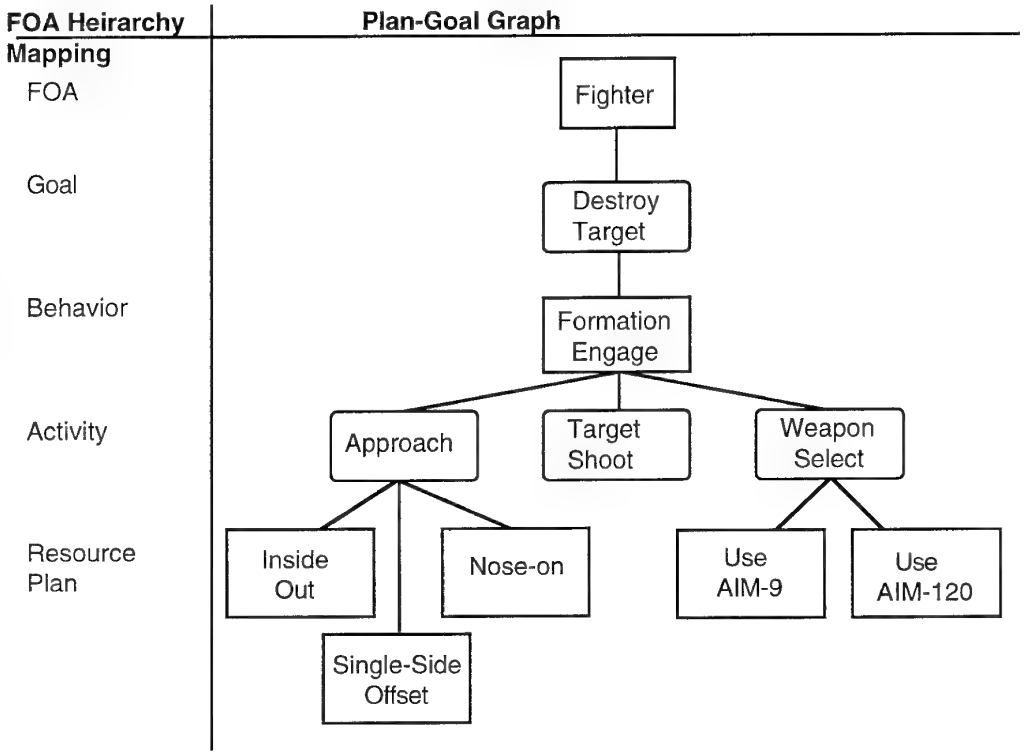


Figure 6.10 Focus of Attention Hierarchy

The top, or "FOA" level, represents the object or event that motivates the formulation of a plan. For example, an enemy aircraft or missile launch would be represented at this level. The second, or "Goal" level represents desired world states that the planner will attempt to realize. Note that a number of Goals may be associated with any particular FOA. For example an enemy aircraft FOA may promulgate distinct Goals for the destruction, diversion or avoidance of that particular aircraft. The subsequent or "Behavioral" level provides for the representation of strategies for the accomplishment of the Goal. As in the previous case, a number of Behaviors may be found to satisfy any given Goal. At the next level, "Activities" supply collections of resources and information about their relationships - dependencies, sequences, synergies, etc. - to Behaviors. The lowest, or "Resource Plan" level, contains elemental plans tied directly to the basic resources - sensors, weapons, systems - of the host aircraft.

The Tactics Planner design is composed from three conceptual elements. First are a set of VxWorks *tasks*. "Monitor task," responds to event signals from the two assessment sub-systems (SA and SS) and PVI. Five additional tasks, one at each level of the FOA hierarchy, initiate reasoning at that level. When a task runs, it executes logic in objects known as *handlers*. Because handlers are objects and may be passed between processes, they form the essential means of communication between tasks. For example, an FOA directs a particular subordinate Goal to specialize its attributes by passing a handler to it. *Reasoners* are the third conceptual element in the Tactics Planner. Reasoners are rule bases for specializing particular classes of FOA's, Goals, Behaviors, Activities, or Resource Plans. For example , all Fighter FOA instances are specialized by the same reasoner which contains the essential knowledge about fighter aircraft.

Control flow in the Tactics Planner depends heavily on the actual situation to which the Tactics Planner is responding. Indeed, the need for great processing flexibility drove the decision to partition control logic into various handlers. Thus, it is difficult to describe a single control path through the system. However, a representative case is offered in Figure 6.11. In this case, the firing of some monitor unblocks a Tactics Planner "monitor task" which then executes to determine the type of processing that will be required. The appropriate "task" in the FOA hierarchy is signaled and a handler is passed to that task. The FOA hierarchy task unblocks and invokes the reasoners needed to perform the requisite plan specialization. This, in turn, may result in signals to other tasks in the hierarchy and subsequent activity.

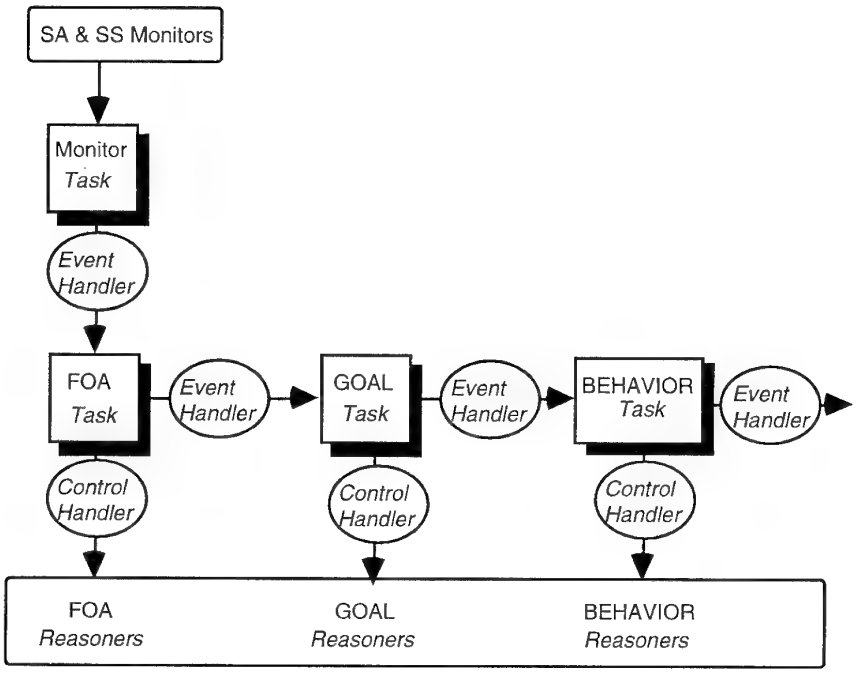


Figure 6.11 Control Flow in the Tactics Planner

This design offers two features that facilitate real-time tactical planning. First is the clear distinction between assessment and planning. Assessment takes place exclusively in the two assessment sub-systems and PVI. Tactical planning is accomplished solely in the Tactics Planner and always occurs in response to the firing of a well-defined monitor. Thus planning activity is always focused and controllable. Secondly, domain knowledge and execution logic are separated and represented in distinct, specialized data structures. Handlers provide control and ensure efficient, speedy and reliable processing. Reasoners apply domain knowledge to defined problems and provide for "intelligent" solutions.

The Tactics Planner currently provides and, sometimes supervises, execution of tactical plans covering a range of tasks from sensor actuation and employment to defensive measures to air-to-air attack of multiple enemy aircraft. The system meets the demands of operational pilots in a realistic air combat simulation and appears to add value to their decisions. Further development of this or similar systems is anticipated in the near future.

6.1.2.4 Summary

The Pilot's Associate provides a good example of the application of the potential application of knowledge based technologies to guidance and control problems - particularly those requiring decision support systems for human operators - in modern military aircraft. The system, as it currently exists, meets practical requirements for real time performance and operation on lightweight multi-purpose avionics processors. It has been applied to genuine operational problems and provides good answers. Pilots are generally pleased with its performance. Quantitative measures of value are currently not available, and given the limited sample size in the test program, it seems doubtful that substantive conclusions will be reached as a result of this program. It is, however, expected that sufficient trend information can be developed to indicate the further development of this technology.

6.1.3 Terrain Referenced Navigation Terrain Following System

The terrain following system discussed in this section forms a part of the covert Terrain Referenced Navigation (TRN) system selected for the Tornado mid life update. This on-board system not only has to satisfy the requirements of real-time operation but is also required to interface with a conventional automatic flight control system in what is generally termed a high-integrity safety-critical application. What is of primary interest is not the standard of the KBS technology content, but the fact that it does contain a KBS element and has been subjected to the full software generation process for safety-critical applications.

In order to set these terms in context the following definitions are offered:

safety-critical – the safety of the aircraft is dependent on the equipment. Usually applied to flight phases with an imminent risk of ground collision, or to situations resulting in loss of control.

integrity – a measure of operational correctness. It can be applied to the effectiveness of the design of the control requirements, to the implementation of these requirements in the software, and to the effectiveness of the hardware in executing the software in a failure free manner. In a safety-critical application the safety of the aircraft is dependent on all three measures.

availability – primarily related to the equipment being in an operable state.

The system architecture exactly matches the functional levels identified in Section 2.2.1 and Figure 2.11. In Section 2.2.1, the concept of functional levels was introduced without specifying whether, in implementation, it would be carried out manually or by machine. In the application described in this section, all levels are automated. The aircraft and state control loop represent the existing avionics. The flight path control loop is a new automated system based on conventional control technology and the flight management loop is a new automated system which contains the terrain following trajectory planning system. It is this element that is of primary interest here, since there is a KBS controlling the formulation of the trajectory to be followed.

6.1.3.1 The Trajectory Planning Problem

The specification of the requirements for an optimal flight path trajectory is relatively simple. It shall be mathematically continuous in its location and derivatives while achieving the lowest possible ground clearance without violating the requirements of minimum ground clearance or maximum aircraft maneuver limits. The optimum trajectory lies somewhere between the unconstrained path which follows every nuance of the ground and the level path above the highest ground feature in the area. The requirement that the system shall design and construct an optimal trajectory introduces new concepts to the specification of avionics functions. The implications of logical planning and decision making within the system are not compatible with the traditional approach to the design of high-integrity safety-critical avionic functions.

A major part of the normal design process is the checking and verification, by analysis and simulation, that the system will correctly function throughout the operational envelope. This results in a requirement specification for a system which will operate with a known level of performance in that it will produce a high-integrity and robust solution. Thus the design process has reduced the task of the system to applying a defined and ordered set of operations to the input data. This is termed control processing and although different paths may be invoked as a consequence of the applied data set, a finite program flow map of possible paths has been designed and only one of these paths will be invoked in the knowledge that it will generate a solution. The set of generated solutions has been designed to be within the set of acceptable inputs to the following process and can therefore be used directly. The probability that it will be an acceptable, or correct, solution is the measure of the integrity of the system. Fly-by-wire control, autopilots and stabilization systems fall within this class. Since they directly affect the flight path of the aircraft by deflection of the control surfaces, they also belong to the safety-critical class of systems. Most avionics systems of this class have distributed frame based structures in order to ensure hard real-time operation at a constant repetition rate. Analysis of the trajectory planning problem showed that, whatever the applied technology, it would be extremely unlikely that it could be formally proven, to the level demanded for safety-critical systems, that an acceptable geometric solution could be guaranteed. Geometric considerations include the effects of location uncertainties and database accuracies as well as the effects of pilot actions.

The first major concern is therefore related to system integrity. Since the system may be placed in a situation where there is no possible solution that exhibits all of the necessary attributes of an acceptable solution, the inherent integrity associated with the traditional control processing design strategy is lost.

The construction of an optimal flight path trajectory through a multi- dimensional terrain model is a complex process. This complexity is further compounded when it is required that the trajectory satisfy a further series of performance related constraints; a variable weighting to minimum vertical clearance, a pilot acceptance of minimum horizontal clearance, a variable pilot acceptance of maximum maneuver levels, and a dynamic uncertainty as to the aircraft location. This form of nonlinear two-point boundary-value problem is representative of the class of algorithms known as 'computationally hard' and more formally defined as 'NP-complete', being characterized as having exponential order time complexity. While it may be possible to construct an optimum trajectory by recursive dynamic programming techniques, the commercial constraints of size, weight, and power in the military aircraft environment are factors which promote the use of technologies which maximize the efficient utilization of on-board processing capabilities.

The second major concern is therefore related to the size of the problem task in terms of the computational power required to generate and maintain a solution in a hard real-time environment.

6.1.3.2 The Trajectory Planning Solution

With respect to the first concern dealing with system integrity, since the trajectory planning process can not be designed to guarantee an acceptable result, a different strategy must be invoked. The fact that a process does not produce a solution which exhibits high-integrity properties, by virtue of having been subjected to the control process design procedure, does not necessarily mean that the process itself can not be high-integrity. Such a class of system is termed analytical processing, which is defined as any data reduction/conversion process which produces a result based on the information content of the applied data set. In this case the computed result may be that there is no acceptable answer based on the applied data and, although not very helpful, represents a failure free high- integrity result. Put another way, the solution set has different dimensions to the acceptable input set for the subsequent process.

In order to maintain the overall system integrity, there must be an additional element which monitors the quality of the solution and responds to the situation where no solution is available with a safe, high- integrity control strategy which will be invoked in this situation. System disconnect, which removes availability, is not a desirable first strategy, but may be the eventual safe strategy. Monitoring is most frequently associated with the detection of failures in hardware elements, however, there is no reason why it should not apply to an assessment of the quality of the processing results, provided that the necessary attributes of an acceptable solution can be formulated.

The trajectory planning system was therefore constructed around a high-integrity central core which monitored for, and managed, this 'soft failure' with a default control strategy which was both demonstrably safe and had the added advantage of simplifying the subsequent trajectory generation process. This architecture is illustrated in Figure 6.12.

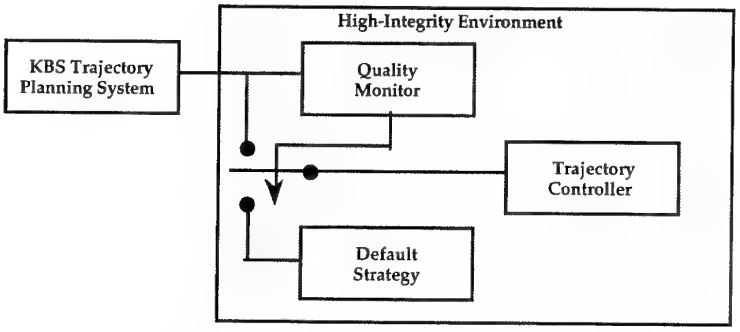


Figure 6.12 Trajectory Planning System Architecture

The second concern regarding computational power requirements is best addressed by the use of efficient algorithms and programming techniques. The hard real-time requirement does not favor good 'average solution times' since the excess time associated with below average values can not be carried forward to ease the above average values. It is the situation where the time allowance is reached without achieving a solution which is of concern. This fact has previously provoked a negative reaction to KBS, and AI systems in general, within the G&C community. Claims that Expert Systems were capable of 'generating solutions for 90% of situations using only 10% of the CPU power' have evoked concern as to what happens in the 10% of situations where a solution is not found. In a fixed frame-rate control system, there must be a solution, or plan, available at each iteration. If the consequence of a 'no solution' result from the KBS system is that a conventional solution must be subsequently generated, then the total CPU power that must be made available, in the worst case, is 110% (10% for the KBS which failed + 100% for the subsequent conventional solution). This requirement to consider worst case conditions will always favor the conventional algorithmic solution.

Since the trajectory path controller was designed to be tolerant to the lack of solution case, the consequences are alleviated to some extent and, in this circumstance, the potential benefits of a KBS solution can be utilized. In very simple terms, the time benefits of the KBS approach are best illustrated by comparison against a conventional structured programming approach. The conventional approach would be to identify a problem and then apply a repair strategy. This would be applied recursively until no more problems could be found. Pseudo-intelligence may be introduced by applying a hierarchical order to the types of problem being sought and also to the order that repair

techniques are applied. In the KBS solution a different strategy is applied. The problems are still sought but they are not acted on immediately. Instead, they are assessed and ranked and a cumulative problem magnitude value is assigned. The assessment is important in that it considers the possibility that the apparent problem may actually be the symptomatic consequence of a different problem and this should be tested before any remedial action is taken. When the problem magnitude becomes significant, or the number of outstanding problems becomes excessive, or all diagnostic processes have been applied, then the scheduler assesses the problem list and invokes the system to solve the big problems first. In repairing a major problem, many minor problems and indirect consequential problems disappear without ever needing to be addressed directly. It is this reduction in inefficiency, by not imposing a rigid flow structure, that affords the timing benefit. An added bonus is that the quality of the final trajectory is often improved. This is a direct consequence of not applying local fixes to the trajectory to solve apparent problems which might subsequently disappear.

6.1.3.3 KBS Description

6.1.3.3.1 Overview of KBS Technique

A classic blackboard approach to the organization of large KBS problems was adopted. This concept is relatively simple and is representative of the class of AI-systems referred to as expert systems (see Section 4.4). The entire system consists of a set of independent modules, referred to as Knowledge Sources (KS's), that contain the domain-specific knowledge within the system, and a blackboard which represents a shared data structure to which all the KS's have access.

When a KS is activated, it examines the current contents of the blackboard and applies its knowledge to add to the contents of the blackboard or to modify the existing contents. Although the execution of the total process consists of the asynchronous execution of a collection of KS's, the execution of an individual KS is a sequential process. Once a KS has been activated, it executes without interruption until it completes. During the execution of a KS, triggers, or demons, may be fired which create activation records describing that a different KS needs to be subsequently activated, together with the reason and the event that fired the trigger. This information can then be used to assess the potential benefit and also to focus the attention of the KS when it is actually activated. Execution of one KS may result in the firing of several triggers, causing several activation records to be created. It is the task of the scheduler to assess the activation records and decide which KS should be applied next. The scheduler is itself a KS which uses knowledge about the capabilities of the other KS's together with ratings of the expected benefits, passed up from the independent KS, to assess which of the KS's is likely to result in the greatest progress towards a solution and should be next activated. When the scheduler finds no pending activation records, an acceptable solution has been found and the flight path trajectory is made available to the path following controller.

6.1.3.3.2 KBS Trajectory Planning

For the trajectory planning problem, five separate domain experts are provided to help in the creation of an optimal trajectory. These domain experts are essentially diagnostic and each has a specific myopic focus against which it assesses the quality of the trajectory and either accepts the trajectory or creates a series of problem reports and associated recommendations. These recommendations could be changes to the constraints, other processes to be activated, or both. Problem reports without recommendations are not permitted. The diagnostic domain experts cover:

ride control	- to minimize bumpiness commensurate with terrain characteristics.
clearance	- to maintain, but minimize deviations above, the set value.
safety	- to maintain safe roll-out recovery tracks.
performance	- to maintain target levels of maximum airframe performance.
continuity	- to maintain kinematic continuity between trajectory sections.

A final process is provided which performs the actual construction, or modification, of the trajectory in accordance with the directions and constraints provided. This process itself contains a simple expert system which activates, from a library of construction techniques, that most suited to the required repair task.

At each invocation, the scheduler assesses the current status of the trajectory, in terms of the list of problem reports, and decides which of the processes will afford the greatest benefit. This decision may be to invoke the repair process immediately or it may be deferred pending further diagnostic assessment.

When all of the diagnostic domain experts have been activated and there are no outstanding problem reports, the trajectory is accepted. All points on the final trajectory will satisfy the kinematic equations of motion and the global

constraints on maneuverability. In addition the final trajectory will exhibit all the properties necessary to satisfy the requirements of each domain expert. It is the dynamic balance of the intermediate local constraints which controls the final flight trajectory.

If the real-time clock mandates a time-out situation prior to a solution being available then the blackboard is reset to correspond to a new set of initial values for the next iteration and a no-solution flag is set.

The choice and use of languages and high level utilities is more relevant to the development phases. In the production phase, the Equipment Specification is likely to dictate the language to be used. It is in the rapid prototyping and development environments that high level utilities can provide the greatest benefits. During the development phase, the desire to fly the laboratory development system in a real-time environment restricted the choice considerably. KBS shells and languages were rejected at the time since none could meet the real-time requirements and none were considered to be sufficiently mature. Also, none were considered capable of giving the required visibility for the flight demonstration. FORTRAN- 77 was used throughout the development phase for the host microprocessor because it represented the greatest maturity by virtue of its popularity and near universal usage as a general purpose language. The visibility afforded by a high order language coupled with the confidence in the compiler proved to be invaluable in achieving rapid software modifications in the flight demonstration phase.

6.1.3.4 KBS Interface with Existing Technology

6.1.3.4.1 Impact on System Architecture and Monitoring

In the design of a conventional, safety-critical, real-time system, the processing load is distributed across the time-frame structure. In the case of a multi-processor environment, this allocation also considers degradation and integrity by grouping interdependent processes and segregating monitors from the processes being monitored. Any loading which is found to be excessive results in a redistribution of modules within the frame structure. Typically an average utilization of 70% to 80% can be expected with no frame overruns.

The KBS element is not compatible with this tightly defined frame structure. The process flow is self determined and, therefore, can not be preassigned across a frame and subframe structure. In order to sensibly include the KBS trajectory planning task within the system it was necessary to allocate a dedicated and uninterrupted block of assigned processor time to the total KBS task. The assessed variability in the processor throughput required to achieve a solution ranged from near zero, in the trivial case, to several MIPS under the most demanding situation found (not necessarily the worst possible). The distribution of throughput exhibited extreme skew and it was necessary to allocate a finite processing power (MIPS) to the KBS element and accept the probability of failing to reach a solution within the allotted time. A nominal 98% success rate was set which resulted in a mean execution time of only 20% and a mode value of about 15% of the execution time allotted to this task. This low average utilization of available processing power is undesirable in an application where power, weight, and volume are all premium assets, however, it represents an application specific choice of a point on the utilization vs. success trade-off curve. It must also be assessed against the processing power that would be required to perform an equivalent function using conventional technology.

In a conventional technology system the assigned processor loading will remain within known bounds and, as part of the hardware integrity monitoring, this attribute can be used to monitor each frame for unusual execution times. A watch-dog timer can detect short or long frame timings associated with failures of the frame executive controller.

This attribute is lost in the design of the KBS element of the real-time system. The variability in execution time precludes the use of frame-time monitors to detect frame-time failures since frame times from near zero to time-out are now acceptable events.

The fact that the KBS element is incompatible with both the system architecture and monitoring associated with conventional technology high-integrity systems supports the segregation of the KBS element, either within a single processor or to its own separate processor. The input to the high-integrity controller is then treated as any other input in that its validity flags are checked together with reasonableness and quality monitoring as appropriate.

Although requiring a more complex failure management philosophy, together with a default real-time control strategy, this form of soft failure management is no more demanding than that associated with a simplex sensor such as a radio altimeter, which is subject to intermittent unlock and loss of availability.

6.1.3.4.2 Impact on Software V & V

The project was managed and organized as any other software engineering project where internal Company procedures are appended to any Equipment Specification requirements. In this instance, the Specification requirement was for all software to be written in a high level language and to satisfy, among others, the requirements of DOD STD 2167A. These requirements are intended to ensure that designs are robust, efficient, and maintainable.

Additionally they require that the system be tested at all levels, from module through integration to full system performance, together with static and dynamic code analysis. The requirements directly related to the levels of documentation and production of code are clearly applicable to any application and cause no specific problems. Those related to testing, however, were not so straightforward in the case of the KBS application.

Basic module testing against the module requirements was problem free. The KBS elements are only in embryo form and can be specified, coded, and tested as any other software module.

Integration testing was more demanding. The KBS element is fully operational and is exposed to the full test procedures. Code analysis using current software engineering tools proved difficult and time consuming. The toolset is designed to identify, among other things, dead code and inefficient constructs. The major difficulty was identifying test harnesses which invoked all the rules since the analysis tool would reject uninvoked rules as dead code.

Performance testing was carried out by closed loop operation and statistical analysis, and also comparison against results from the original VAX-based development model. At this level the KBS element is treated as a closed function and presents no specific problems.

6.1.3.4 Status

The development system has been demonstrated on a variety of UK, US, and European airframes. This has resulted in the selection of the terrain following subsystem for the Tornado mid-life-update program. Based on the Specification for this system, a production standard equipment has been developed and tested against typical current requirements for safety-critical avionic systems.

6.1.3.5 Summary

The safety of the aircraft is dependent on the integrity of the design process, the software implementation, and the host hardware. In a conventional avionic system the design process is checked and verified by analysis and simulation throughout the operational envelope. Tolerance margins are included and sufficient linearity is assumed that operation will be possible between the design points. This results in a high-integrity and robust requirement being passed to the software engineer. He operates within the framework of Standards and Procedures to implement the requirement in a correct manner while adding monitoring and BIT and testing to ensure the operation of his product against the requirement. Similarly, the hardware engineer, by monitoring, selection, and replication, ensures the availability of a failure free host environment.

In a KBS environment, the hardware engineering task is unchanged. The software engineering task is also unchanged, it is the software requirement that is different. The requirement is now specifying a system which will carry out the design task and also perform the resultant control requirement. While it is possible for the design engineer, the software engineer, and the hardware engineer to all impose the highest quality or integrity on their products, it is difficult to assign an integrity value to the quality, or even existence, of a solution from the KBS element which is now carrying out the design, or planning, task. It is therefore important not to confuse the integrity of a process with the acceptability, or integrity, of the result of that process. This is dependent on the mapping of the possible output set from one process onto the acceptable input set of the subsequent process.

To maintain the safety of the system, this reduction in the system integrity must be compensated for by the introduction of the premise that the KBS element output is of low integrity possibly with an associated failure rate. There is now a requirement to both monitor the process output and also invoke a default recovery procedure in the event of a failure. This has been achieved by the provision of an independent monitor on the KBS element which checks the solution for completeness and constraint violations. A soft failure is flagged against any non-compliant attributes and a default recovery control strategy is invoked. If the soft failure condition persists, then as a function of time and maneuver, it converts to a hard failure status.

Software engineering standards and procedures dictate comprehensive requirements to ensure the generation of high quality software, against the requirement specification, for safety-critical applications. The use of formal methods and software toolsets is encouraged while standards of documentation, traceability and accountability are mandatory. These standards have been developed and proven against a class of safety-critical avionic systems that can best be described as 'conventional'.

By separating the attributes relating to the conventional element and the KBS element from the quality, or acceptability, of the solution, it has been possible to apply the same Standards and Procedures to both elements. An avionic system containing a KBS of the expert system, or rule based, class has been successfully developed for a safety-critical application against these standards without any need for change.

There is, however, a need to ensure that the KBS element does not compromise the system integrity for safety-critical applications. In this application, this has been successfully achieved by the segregation of the KBS element and the provision of an independent monitor which checks the solution for completeness and constraint violations. This normally will imply a strategy leading to the eventual disconnection of the equipment, thus maintaining the integrity of the system at the expense of availability.

6.1.4 COMPAS - Computer Oriented Metering Planning and Advisory System [6.29-6.33]

6.1.4.1 Introduction

The COMPAS system is an example of the successful introduction of knowledge-based planning support in Air Traffic Control. The basic functional concept of how the computer-based functions for *Monitoring, Diagnosis* and *Planning* support the other mental control functions of the human air traffic controllers have been outlined in Chapter 2.2.3 .

COMPAS is a planning tool to assist the controller in the handling of arrival flights in the extended terminal area of major airports. It aims at improving the flow of traffic and the efficient use of the available airport landing capacity while reducing planning and coordination effort of ATC personnel. The system has reduced controller workload of the approach controller team and does not cause any significant additional load to the en-route controller teams.

Main basic functions of the system are *Monitoring* and *Diagnosis* of the traffic situation based upon the on-line input of the initial flight plans, actual radar data and the actual wind. Basic *Planning* parameters such as: aircraft performance data, air-space structure, approach procedures and controller strategies, separation standards and wind models are already stored in the computer and do not require additional inputs by the controller.

Each time a new aircraft enters the predefined planning area, COMPAS determines the optimal sequence of all approaching aircraft and calculates a schedule of arrival times for the "Metering Fix", a waypoint at the Terminal Maneuvering Area (TMA) boundary and the "Approach Gate", a waypoint on the runway centerline. The computer-derived optimum sequence and schedule and some advisories on achieving the desired plan are displayed to all Controller teams who are responsible for the control of the inbound traffic. Each of these teams receives only those data which are necessary to control the arriving flights in its sector and to contribute to the optimized overall plan. Usually there is no interaction required between COMPAS and the *Human Operator*. However, the controller has the ability, if he sees the need, to modify the computer generated plan or to change planning parameters and constraints through a small function keyboard.

6.1.4.2 Monitoring, Diagnosis and Planning process

The whole process is divided into several steps:

- the acquisition and extraction of radar and flight plan data (**Monitoring**)
- the prediction of the flight profile and the calculation of the arrival times (ETO) as if the aircraft were alone in the system, checking for time-conflicts at the so-called "Gate" (**Diagnosis**)
- planning of the optimal overall sequence and calculation of the planned arrival times with minimum total delay for all known inbound flights (**Planning**)
- freezing of the planning status when the aircraft reaches its top of descent.

There are several assumptions within the flight profile model with regard to an economical descent profile and the performance of the type of aircraft. A simplified method based on airline operations data was developed for profile prediction. The different profile legs are calculated with the actual radar position, airspeed, flight plan data, altitude, wind data as received on-line from the ATC data processing system. Further consideration is given to the aircraft type-specific economical descent gradient, minimum cost descent speed, the aircraft deceleration rate and possible air traffic constraints at the Metering Fix and the Approach Gate. The estimated time of arrival (ETO) is based on the preferential flight profile of the aircraft . The *earliest* estimated time of arrival (EETO) takes into account all measures to advance the aircraft within its performance envelope without requiring any thrust increase. The time difference between ETO and EETO is used as a margin for sequence changes to maximize traffic throughput without violating economical flight conditions.

With its EETO, the newly entering aircraft is inserted into the already existing sequence (see Fig. 6.13) The result is an initial plan, i.e., a tentative sequence of aircraft according to the 'first-come- first-served' principle, but possibly with still unresolved time-conflicts.

As an example of the knowledge-based core functions in COMPAS, the planning algorithm to establish the optimal sequence and schedule shall be described briefly . It is an analytical 'Branch-and-Bound' algorithm with three major elements.

- o merging of new arrivals into the sequence
- o time conflict detection and
- o time conflict resolution with optimization criteria.

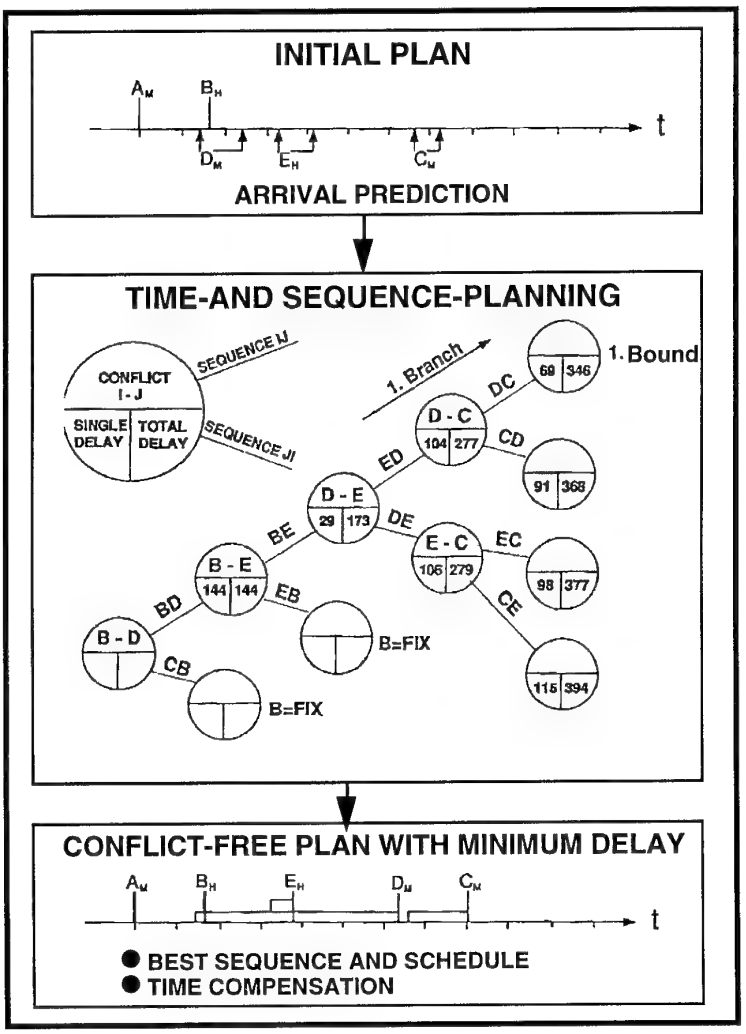


Figure 6.13 COMPAS Planning Algorithm

The overall goal here is to minimize the total delay time by optimal combination of the aircraft of different weight classes. A dense sequence of aircraft (i.e., minimum total delay) contributes to the best utilization of the available runway capacity. The algorithm can be graphically represented as an heuristically oriented search in a decision tree. The nodes represent the individual sequence pairs which are characterized by the earliest time conflict between two aircraft in each case. The branches show the alternatives for conflict resolution and the decision tree is developed following the principle 'solve-the-earliest- conflict-first'. The cost function is the total delay time, which is accumulated until a conflict free plan is found. The cost value of this first solution is called 'first bound' (usually

it is a sub-optimal sequence). A backtracking procedure leads sequentially to all those nodes with less then the total delay of the first bound. From there new branches in the search tree are developed. The development of a new branch will be stopped either when the total delay value of the 'first bound` is exceeded or it leads to a new bound with less total delay. The planning process ends when all remaining conflicts have been resolved. The result is a sequence for all known inbound flights with the shortest time separation between any preceding and trailing aircraft equal or greater than minimum permitted separation and a planned delivery time for all arrivals at the so-called "approach gate". From this "gate time" all other intermediate arrival times for other waypoints are calculated individually for each actual flight. Furthermore, an advisory is calculated which defines how much each arrival has to be advanced or delayed.

6.1.4.3 Display

The layout of the man-machine interface of COMPAS was of crucial importance to the acceptance of the whole planning system. "Keep the controller in the loop". "Give him the plan, but leave the implementation to his experience, skill and flexibility". "Minimize the need for keyboard entries and keep the advisories as simple as possible". These were the main guidelines and principles for the design of the Human-Computer interface, i.e. the COMPAS-display and the COMPAS-keyboard.

The display of the solutions from sequencing and scheduling is specially tailored to the needs of the different ATC units (Enroute and Approach) which are involved in the handling of inbound traffic. Figure 6.14 shows the features of the controller display. The planned aircraft in the arrival flow are listed sequentially in a vertical time line, with the earliest arrival at the bottom. The aircraft labels are additionally marked with 'H' or `L` according to the weight categories of the individual aircraft (H=HEAVY, L=LIGHT, the standard MEDIUM category is not indicated explicitly). The vertical position of each arrival is correlated with a vertical time line which moves downward with the progress of time. The bottom of the line represents the planned time over the Metering Fix (enroute display) or Approach Gate (approach display). A color code is used to indicate from which approach sector the aircraft are coming. The letters left of the time line give a rough indication (advisory) of the suggested control action for each aircraft during the descent phase. Four characters are defined in order to reach the planned arrival time and to establish a dense, smooth traffic flow ('X`= an acceleration of up to two minutes, '0`= no action, 'R'= a delay of up to four minutes and 'H' = more than four minutes delay). The controller is free to accept or to reject the advisory. He can modify the computer-generated sequencing plan if he desires or if unforeseen events have occurred.

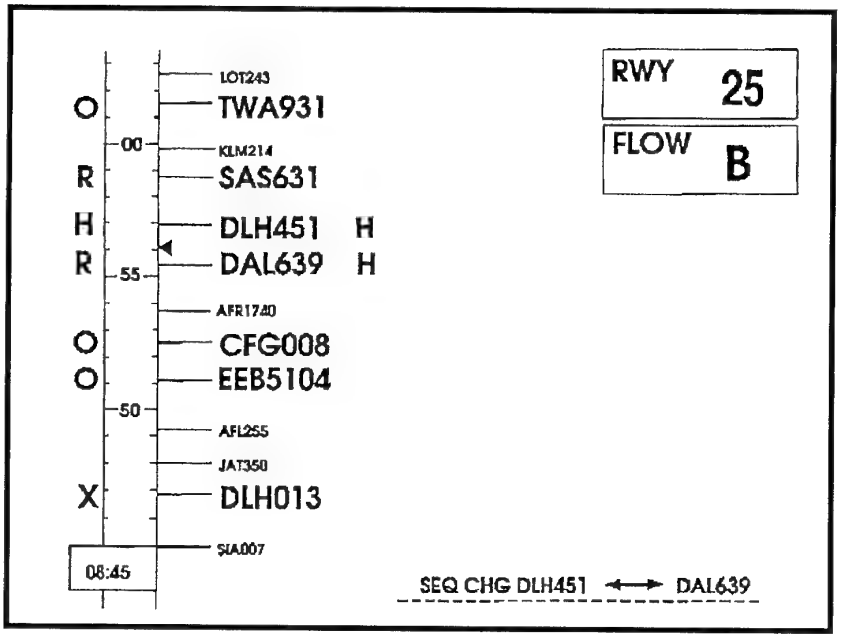


Figure 6.14 COMPAS Enroute Controller Display

In addition the display shows, at top right, two basic parameters for information: the active runway direction (e.g. 25) and the so-called "FLOW", which actually tells the minimum permitted and planned separation.

The controller can move a cursor up or down on the time line to identify a specific aircraft or time window if he wants to enter modifications.

6.1.4.4 Keyboard

Figure 6.15 shows the intentionally very simple functional keyboard for controller-computer interaction. There are ten function keys to change the basic parameters or operational functions.

Inputs to modify the basic planning parameters can only be entered by the approach controller, i.e.:

- o change of minimum separation (FLOW),
- o change of landing direction (RWY CHG) and
- o STOP in case of closure of runways.

Inputs to modify the automatically generated sequence and schedule can be entered at all controller keyboards, both in the in enroute and approach sectors, i.e.:

- o insertion of arrivals unknown to the system into sequence (ADD SLOT),
- o cancellation of planned arrivals (CNL SLOT),
- o move an arrival over several positions in the sequence (MOVE),
- o change of planned sequence between two successive aircraft (SEQ CHG),
- o assign priority to an arrival(e.g. emergency, ambulance, etc.)(PRI)
- o display of additional information to the en-route controller (TFC INFO), to give additional information about aircraft in adjacent sectors.

6.1.4.5 Conclusions

The COMPAS-system was installed in the Frankfurt/Main Air Traffic Control Center of DFS, the German Air Traffic Services in 1989. Since then it has been used successfully, 24-hours-a-day, and has shown improved traffic flow and throughput. It has found overwhelming acceptance with the human operators. As scientific, statistically proven studies have shown, this is mainly due to the planning results which are generated by the knowledge-based functions. They are reasonable, feasible and easy to comprehend and to apply. Controllers feel that these advisories are "non-intrusive", and give an easy-to-follow framework plan while allowing them to stay in the loop with some flexibility to make changes. Above all, the controllers retain the ultimate authority and responsibility.

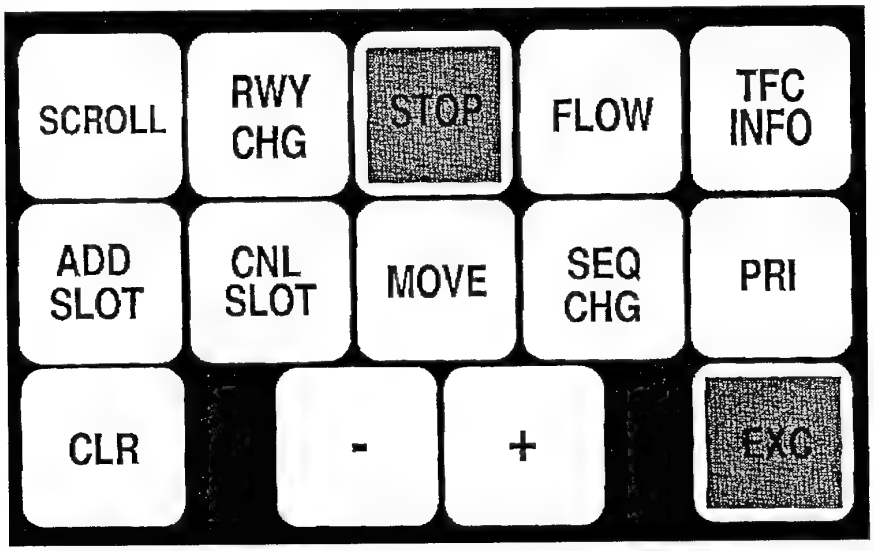


Figure 6.15 COMPAS Keyboard

The future development of COMPAS is focused on two main directions:

- o incorporation of more sophisticated models, advanced planning technologies and heuristic planning methods as described in Chapter 4.
- o full integration with other planning support tools for ATC Terminal Automation, (e.g.: 4D-Planning; Wake Vortex Warning System; Arrival/Departure Coordination; Airport Surface Traffic Management)

This will be achieved through the application of advanced knowledge-based technologies (e.g.: Multi-Agent Planning; Hierarchical Planning; Multi Sensor Data Fusion; Information Management) as described in Chapter 4.

6.2 Space Applications

6.2.1 The Real-Time Data System (RTDS)

The Real-Time Data System project was the first major success within the United States space program in integrating knowledge-based systems technologies into the operational world of space vehicle monitoring, diagnosis, and control. It was initiated in 1987 at the Johnson Space Center (JSC) with the following major goals [6.34]:

- To inexpensively introduce modern computational technology into real-time Space Shuttle operations
- To improve the quality of flight controller real-time decisions by using KBS reasoning methods
- To improve the quality and efficiency of training for flight controllers
- To provide a flight control system development platform external from the operational center which would allow for testing of new computational technologies and easy integration into the operational flight control rooms
- To transfer applicable technology for flight control into Space Station Freedom operations.

From an initial application to the communications subsystems of the Space Shuttle (the Integrated Communications Officer or INCO console), RTDS has now reached the stage where advanced graphics and AI technology has been infused into 11 of 19 flight control positions. These include such critical areas as propulsion, electrical systems, communications, and guidance. The project has also expanded in scope to real-time data acquisition/distribution since there was previously no means of providing telemetry to the engineering workstations on which the RTDS applications are hosted. RTDS has grown to be both an operational platform in the Mission Control Center (MCC) and a development platform in outside office buildings. Perhaps the most significant impact of the RTDS project has been the demonstration, proof, and programmatic acceptance of a distributed, networked workstation architecture for the new Space Shuttle and Space Station Combined Control Center.

The first RTDS systems made use of graphical user interfaces and heuristic reasoning techniques (expert systems) to "translate" the display screens of numbers in traditional control systems into engineering oriented displays. The comparison can be seen below in Figures 6.16 and 6.17 for the INCO console system. This had the effect of decreasing errors and substantially reducing training time to become expert INCO controllers.

Three techniques were used to facilitate the translation from a "flat" display of parameter values into the RTDS displays. First, the major part of the RTDS INCO display is organized as a schematic of the Space Shuttle communications systems. Functional interconnections are easily seen and engineering parameters (both numeric and qualitative) are displayed immediately adjacent to the relevant components. A skilled communications engineer needs very little additional training specific to the INCO console to effectively monitor the system. Second, while it cannot be seen in the figure below, color is used extensively to highlight warnings and potential component failures. Parameters, associated components, and potentially affected inter-component linkages turn red when immediate action may be required. This is nearly impossible to overlook compared to a numeric change in one of hundreds of parameters on the prior system. Finally, a knowledge base of hundreds of heuristic rules is used to analyze and classify parameter changes. Instead of simply showing out-of-limit parameters, the system is able to correlate changes and provide advice on problem nature and criticality. For example, the system can help decide whether a problem has occurred in a single component or in communications links among components by an analysis of parameter changes.

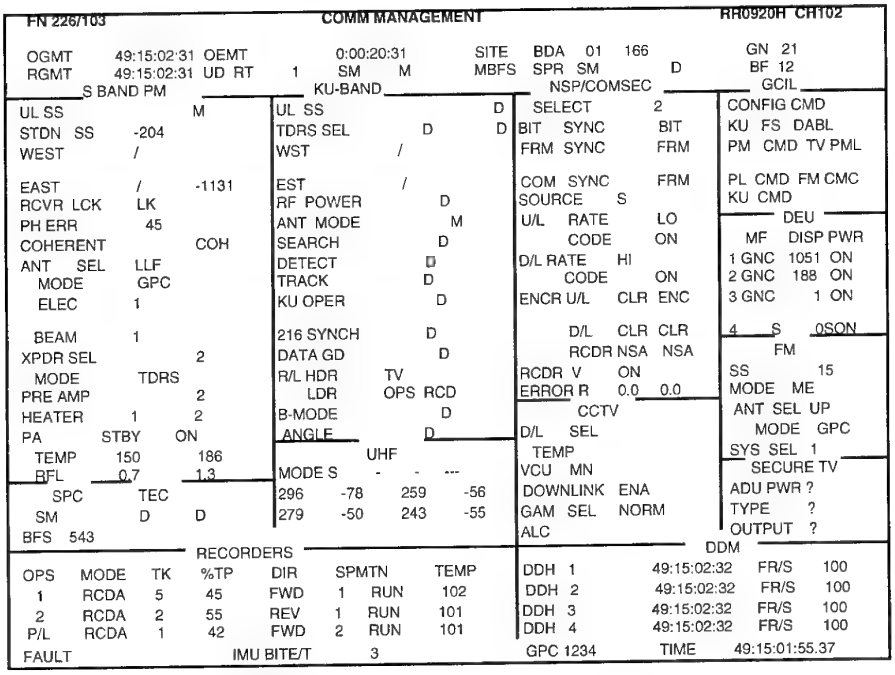


Figure 6.16 Traditional Display of Communications Systems

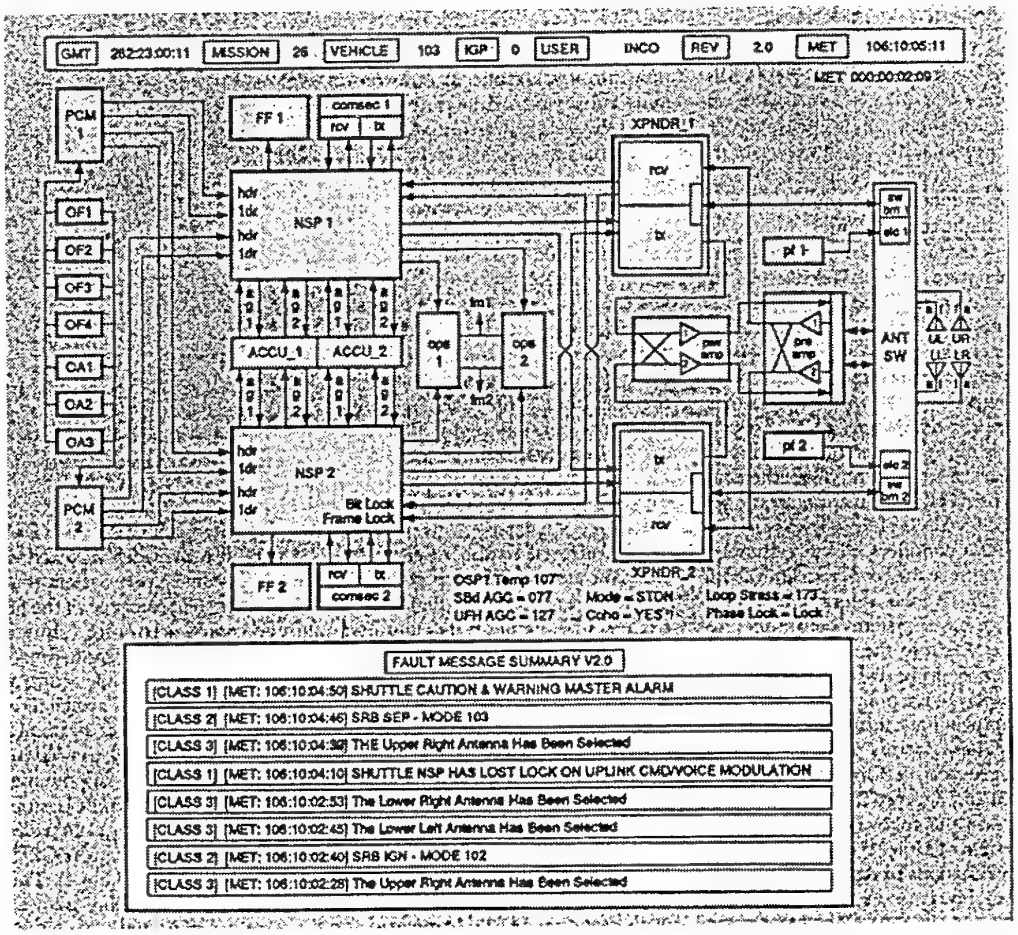


Figure 6.17 RTDS Display of Communications Systems

Within the last several years, "second-generation" AI technology has begun to be used to upgrade existing RTDS applications and provide a core for new systems. These technologies, mainly model-based reasoning and machine learning, are used to correct the major weakness of expert systems: the inability to reason about previously unseen problems. For example, the Fuel Cell monitoring system shown in Figure 6.18 employs a detailed causal model of the electrical components, pipes, valves, and tanks to reason directly about pressure and voltage changes and assign likely causes to those changes. It "understands" failures in the system even if the precise pattern of parameter values has not been encoded within heuristic rules.

RTDS has made a qualitative difference in the way space mission control is being conducted for the NASA manned space program. The shift has been from centralized data processing on a mainframe computer to distributed processing on workstations specialized to each engineering discipline. AI technology provides heuristic and model-based diagnosis techniques to bear in situations where algorithmic methods are either incomplete or too slow.

The technology has also provided substantial cost savings in the Mission Control Center at Johnson Space Center. This has occurred in three ways: by reduction of actual console operator positions, by reduction of training costs, and by reduction of software development costs. The latter has been by far the most dramatic, with a recent estimate of a $160 million cost savings by the year 2000 in creating the Control Center Complex that will be used for on-orbit operations of the Space Shuttle and Space Station Freedom.

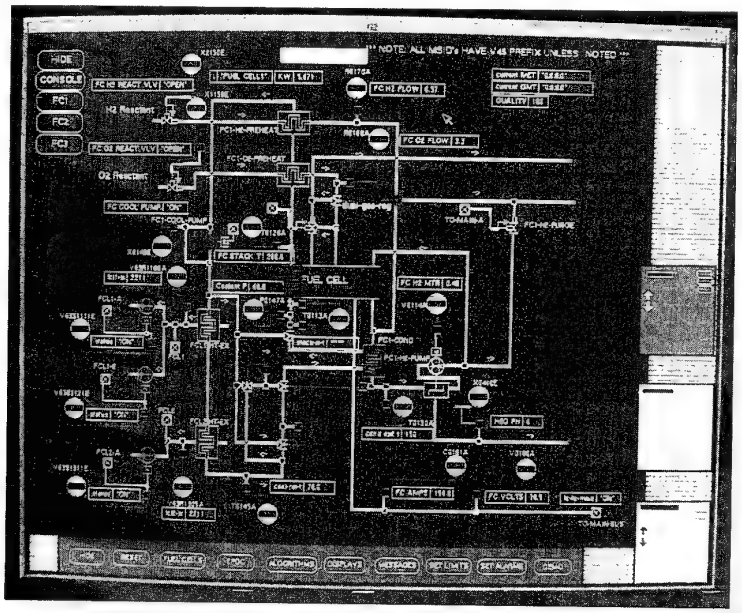


Figure 6.18 Model-Based Fuel Cell System Display

6.2.2 ARIANEXPERT : A KBS For Post Flight Analysis Of Ariane Launcher

ARIANEXPERT is an excellent example of KBS potential for the spacecraft monitoring function (see § 2.2.4), and, more specifically, of KBS capability to reason about physical systems (see § 4.4). ARIANEXPERT has made the Post Flight Analysis (PFA) of ARIANE launcher telemetries, which were recorded during the flight, far more productive (by a factor of 4), while making this task more systematic and more exhaustive, and improving the quality of the task. Furthermore, because of its capability to store PFA's for some time, ARIANEXPERT captures the PFA expertise, and thus provides a technical data base of ARIANE flights and PFAs.[6.35, 6.36]

6.2.2.1 Statement Of Problem/Solution

The ARIANE launcher post mission analysis is done at ARIANESPACE. This activity is called the "Level 0 Post flight Analysis" and is carried out after each launch by about 60 engineers who are working together under the leadership of ARIANESPACE.

The PFA is one of the most critical ARIANE operations, for several reasons:

- the launch rate (8 a year for ARIANE 4) allows only a very short time to carry out all the verification work. Moreover, the PFA is a mandatory step before authorizing the next launch;

- the complexity of the ARIANE launcher results in a very high demand on the PFA engineers. Moreover, there are problems of availability of people with relevant expert knowledge (characterized by a substantial staff turn over during the 10 year life duration of ARIANE 4) which could potentially result in errors or omissions;

- it is very important to be able to take into account the experience of the preceding flights and to record the results and the knowledge accumulated for each launch;

- the quality and the reliability of the PFA mainly depends on the accessibility of data and on the methodology used.

The PFA is composed of the analysis of fourteen flight domains :

- first-, second- and third-stage propulsion domains;
- liquid and/or solid boosters (depending on the launcher configuration);
- electrical systems of the vehicle equipment bay;
- cap mechanical analysis (separation);
- flight program;
- trajectory performance;
- automatic pilot during propulsion phase;
- flight mechanics;
- attitude control system;
- vibrations;
- POGO.

The PFA is still largely done manually, and does not benefit from improved methodologies and advanced technologies providing computerized support for data processing and diagnosis. Consequently, ARIANESPACE has sponsored MATRA MARCONI SPACE for the development of a KBS, called ARIANEXPERT, for supporting the PFA activity. This system combines AI techniques and Numerical Analysis techniques, together with advanced graphical capabilities.

6.2.2.2 Benefits Of KBS Techniques

The AI techniques were used to provide the system with high flexibility capabilities:

- possibility for the experts to enrich the analysis Knowledge in the system by themselves. This was achieved by using an Object Oriented representation of knowledge and reasoning mechanism.

- manual or expert access to the rich set of graphical and numerical capabilities of the system in analysis phases.

6.2.2.3 Interface With Existing Technologies

ARIANEXPERT is interfaced with more conventional software technologies such as a data base on a mainframe that provides the telemetry files or the FrameMaker ^{TM} word processor to automatically format an analysis report.

6.2.2.4 ARIANEXPERT Main Modules

The system architecture is shown in Figure 6.19. The Feature Extractor is the manager of the PFA. It directs the different data processing and plots, handles the fault detection and trends analysis, activates the diagnosis facilities, records analysis results in the PFA database and drives the automatic report editing.

The Man Machine Interface provides the interaction tools between the user and the system.

The Diagnosis Facilities provide the knowledge bases (KBs) and the different mechanisms (KB Methods) to interpret the detected anomalies.

Finally, the KB Administrator is a toolbox containing dedicated editors to enrich or modify the different knowledges used by the system such as diagnosis KBs or fault detection knowledge. It also handles a special mechanism to take into account new fault patterns.

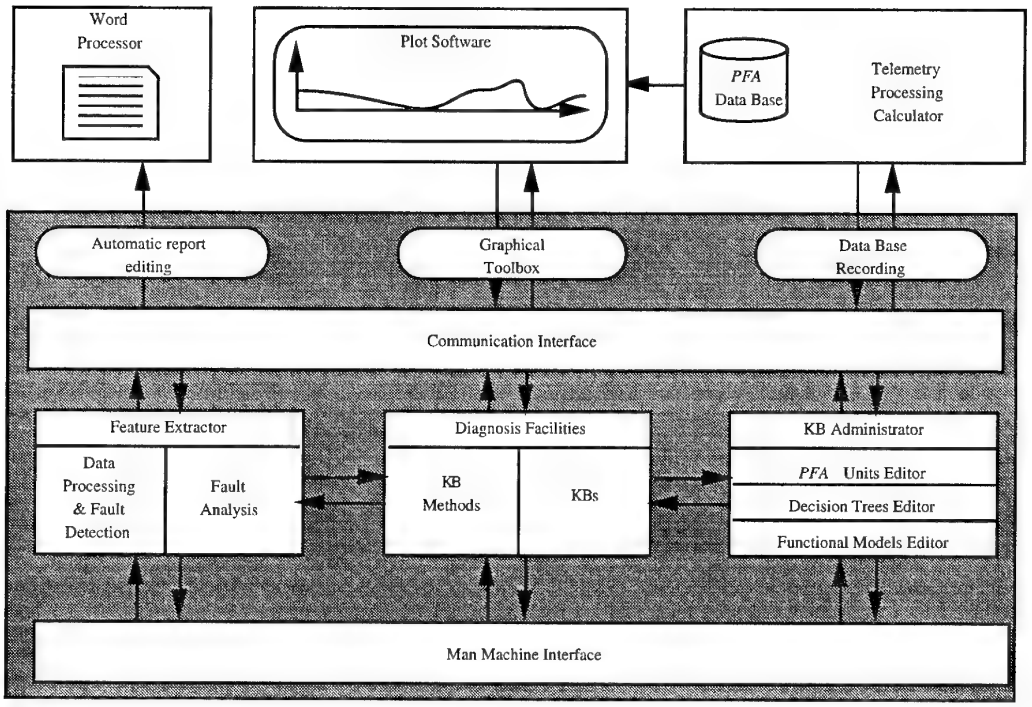


Figure 6.19 ARIANEXPERT General Architecture

6.2.2.5 Status

After a 6 months experimentation phase, ARIANEXPERT became operational at ARIANESPACE in early 1991. The operational deployment of the system was a success. Today the availability of ARIANEXPERT has become mandatory for the users at each PFA. A major benefit is that performing the PFA with ARIANEXPERT requires 1/4th the time than previously and, in addition, ARIANEXPERT allows addressing systematically and exhaustively more analysis than before.

6.2.2.6 Maturation

Introducing such innovative tools as operational systems was a challenge that was successfully met due to the following factors:

- A major factor was the iterative development methodology This limited the technical demand placed on the initial prototype in order to deliver a first usable version to the end users very quickly in order to get their feedback, The final users were kept involved in each step of the development process; feasibility study, mock-up development, prototype development and operational deployment.

- In addition the system was designed to allow an incremental knowledge acquisition and experimentation from the beginning of the prototyping phase.

- Then a highly motivated user was identified to become the administrator of the system. His mission was to handle the evolution of the system by managing the interface between PFA Operators and other Domains Experts as well as to support them in the use of the system.

- Finally, because of the psychological organizational impacts generated by these innovative tools, it was necessary to introduce them very progressively in its industrial target environment.

6.2.2.7 A Typical Analysis Case Performed With ARIANEXPERT

The user loads the knowledge base corresponding to a flight domain and selects the flight. The ARIANEXPERT system then loads automatically all the telemetries which are required for the flight domain and performs successively all the analyses which have been defined in the knowledge base by the experts.

The analysis procedure is simple and always the same. This is an important point because new users can learn very quickly how to use the system. The system first explains the goal and the reasons for the analysis and displays the telemetries involved in the analysis (see Figure 6.20). The analysis consists of a data processing performed automatically by the system on the telemetries. The result of the data processing is compared to the reference values specified by the experts of the flight domain. ARIANEXPERT is then able to detect abnormal values and to warn the user of these anomalies. The user is released from the tedious tasks which are performed automatically by the system and he can concentrate on the on-going analysis.

This simple analysis procedure is powerful because many types of data processing are available in the system and because it is possible to organize the single analyses into a complex conditional analysis procedure. The system explores an exploitation tree (see Figure 6.21) which contains not only single analyses but also tests. Single analyses placed below a test are performed only if the test is positive.

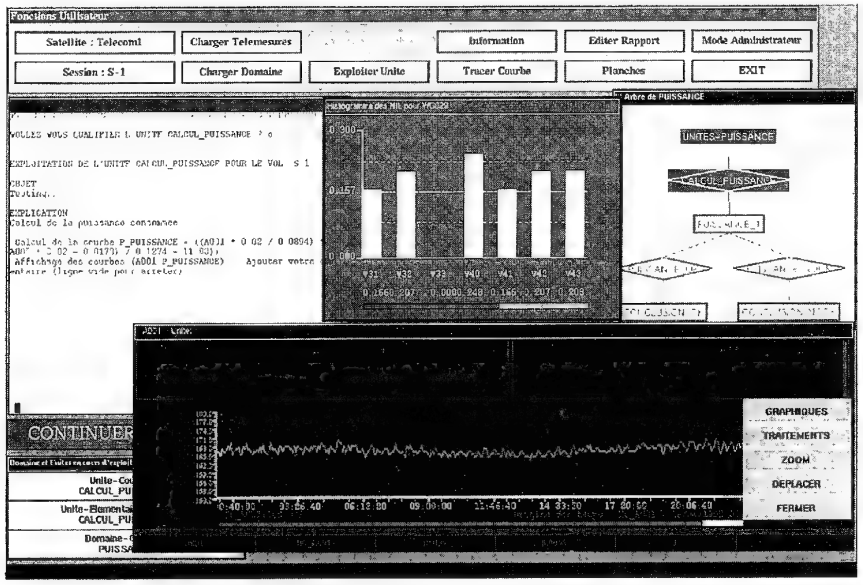


Fig 6.20 Main Window of ARIANEXPERT

The user can suspend the analysis at any time to perform complementary investigations on the data using the interactive graphical functions proposed by the system.

At the end of a working session, a final report on the PFA can be automatically generated by ARIANEXPERT. Several types of report formats can be defined by users, depending on whether they want a synthetic or exhaustive report. The user can also select telemetry graphics during the analysis which will be included in the final report (see Figure 6.22).

The user can find all the results associated with an analysis in the report. The Figure 6.22 gives a report example for the analysis of the offset of the gyroscope during the 5 seconds before the launch at Ho. The mean value of the offset must be lower than a reference value for the three components roll-pitch-yaw. The main characteristics of the analysis are listed first: purpose of the analysis, telemetries which were studied, the analysis time interval and the validity criteria of the analysis. Results are displayed in a three-column table. In this example, the validity criteria is fulfilled for each telemetry, i.e. the observed offset is lower than the maximum reference offset.

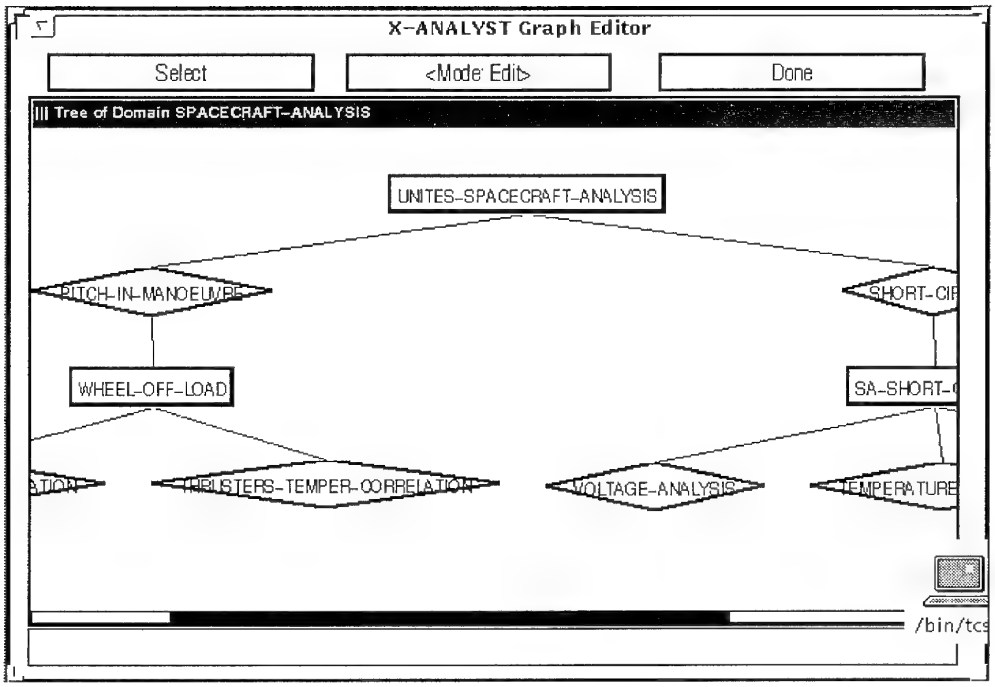


Figure 6.21 Analysis Procedure Tree.

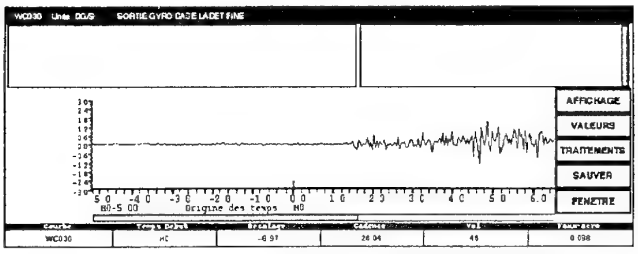
1.1.2 GROUND-OFFSETS.2

Purpose : Analysis of the Equipment Bay fine gyros offsets.
Telemetries : (WC029 WC030 WC031).
Datation : (H0-5 H0)
Qualification criteria : the offset value between H0-5 and H0 must be lower than the reference value for the TM.

Observations :

Telemetries	Observations	References
WC029	0.223	0.3
WC030	0.194	0.3
WC031	0.238	0.3

Comments : Everything is correct.



1.1.3 GROUND-OFFSET.4

Purpose : Analysis of the L220 fine gyros offsets.
Telemetries : (WC026 WC027 WC028).

Figure 6.22 Example Report Page

6.2.2.8 Field Experience With ARIANEXPERT

The main challenge for the successful application of ARIANEXPERT to the PFA was related to human factors. The system had to be used by people with little experience in using computers for interactive analysis and who had their own habits in performing the PFA. It was quite a big change in their way of working and people were rather reluctant during the first working sessions with the system. Reluctance left progressively place to conviction mainly because :

- It is integrating conventional computerized support together with advanced analysis functionalities.
- low-level tasks are automated and data is very accessible
- it is possible and easy to enrich or modify the knowledge of analysis progressively with flights
- it is easy to understand the results of analysis and efficient to have them being formatted automatically in a report
- it is possible to compare analysis results of past flights taking into account the different launcher's versions
- the results given by the system are complete and very precise,
- ARIANEXPERT is very easy to use although it provides a very rich and advanced set of functionalities.

6.2.2.9 Conclusion & Potential For Other Applications

Now that ARIANEXPERT has achieved a good state of maturity, a generic reusable kernel called X-ANALYST, including all the main functionalities of the system has been extracted in order to adapt it to other data and trends analysis problems such as in-orbit spacecraft monitoring., design data analysis, etc. Of course its analysis procedures and diagnosis knowledge representation modules can also cover other data analysis domains, which opens a wide potential for other applications to X-ANALYST.

ARIANEXPERT development was initialized in 1988, and its first operational version was delivered to ARIANESPACE in January 1991. Enhanced versions of ARIANEXPERT have since been delivered.

The X-ANALYST generic kernel was isolated in 1992; its first other application, SAT-ANALYST for satellites trends analysis, was developed and delivered in 1992. It is now being used within several space programs (e.g. TELECOM 2, HISPASAT...). X-ANALYST has been recently adopted for ARIANE 5 (the future version of the ARIANE launcher) test benches, for testing the launcher prior to launch.

6.3 Life-Cycle Example Applications

6.3.1 Management of Life Cycle : Copilote Electronique

This chapter illustrates AI life cycle Management, as mentioned in chapter 2, through the initial phases of a KBS development project. The example here is the description of the early phases of the French Copilote Electronique Program. This program is representative of a multiple-domain, multiple-partner project with a fixed-cost approach. The various efforts described resulted in the establishment of the Copilote Electronique management structure intending to optimize the development phases of the life cycle. [6.37-6.43]

6.3.1.1 Introduction : Problem description

Around 1986, the French Aerospace Defense Services began considering the emerging problem of aircraft mission management being adversely impacted by a growing information and cognitive task Pilot overload. At this time, AI technology was moving toward new decision aids and in the US, the Pilot's Associate program was also receiving more attention in European defense departments.

An action was then initiated by the French military research agency, DRET, to survey the need for pilot assistance in French air force programs and the feasibility of KBS as a potential technical answer. The need was expressed by senior pilots of the French air force and navy, based on experiences with the Mirage F1, Mirage 2000 and Super Etendard. This survey was completed by Dassault test pilots currently involved in the definition of the new Rafale program. A rapid prototyping approach led the Dassault team to mock up a simplified assisting system, which showed enough promise to lead to a more important action of Exploratory Development. For this development a fixed cost constraint was imposed and the need to federate various industrial experiences was expressed.

The history of the Copilote Electronique early phases can be viewed as a typical process that many G&C KBS projects are likely to follow if they involve multiple domains of expertise and several industrial partners. This example covers mainly the initial AI life cycle iterative phases as presented by the "Knowledge Engineering Life Cycle" spiral shown in Figure 2.33 Three main phases are described: the concept feasibility phase; the technical studies phase; and the federative project definition phase. It gives some insight regarding the various actions performed by the French Aerospace partners from 1986 to 1992. These descriptions show how such a project is faced with the development infrastructure needs as described in chapter 5. Ad Hoc solutions in terms of methodological approach and design tools are presented with estimates of their merits (precise metrics could not be issued as unrestricted material).

Figure (6.23) depicts the cycles Dassault and other French Aerospace industries went through before the launch of the exploratory development "Copilote Electronique" by the French Defense Services.

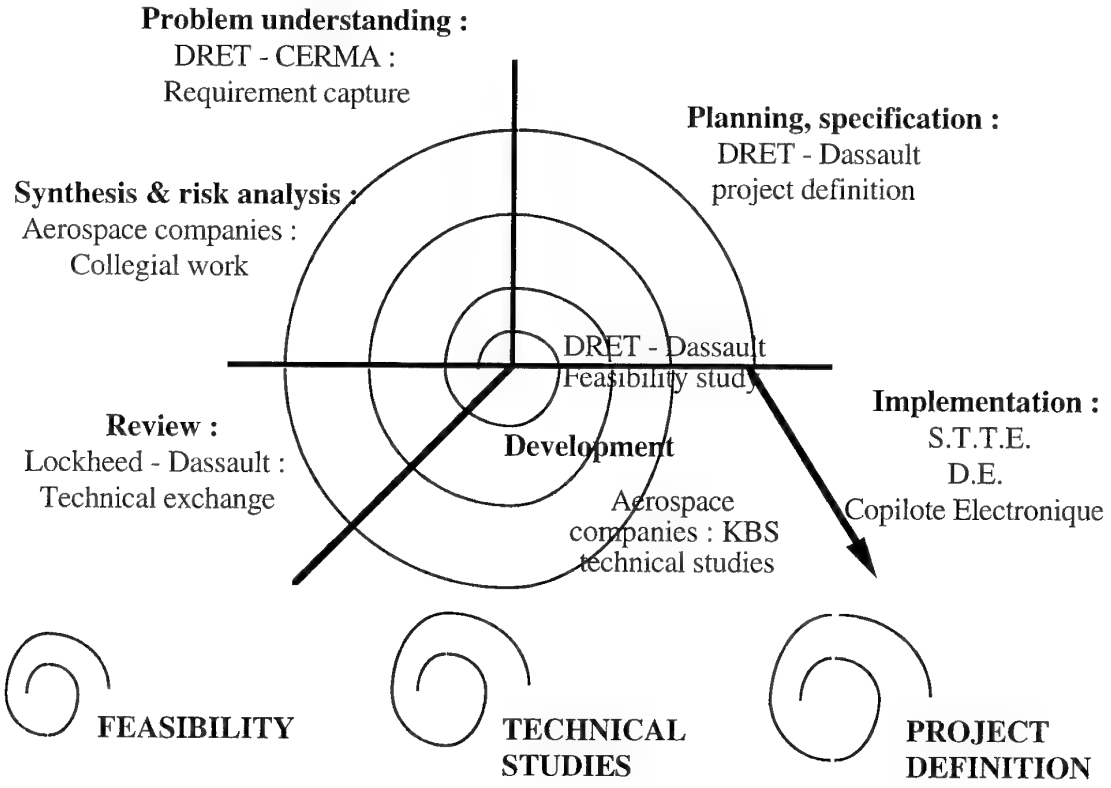


Figure 6.23 Copilote Electronique Initial Phases

6.3.1.2 A first iteration example : the feasibility phase

All fighter aircraft equipment follow the same cycle; better or new techniques and technologies —> improved performances —> increased complexity —> better or new techniques and technologies—> etc. The resulting workload may overextend the pilot's capabilities resulting in a drop in the global performance of the aircraft and may even lead to a mission failure.

After the initial understanding of this problem and the generic needs identification, the French Defense Services launched several actions in parallel through the Aerospace community in order to study the feasibility of an onboard assistant concept.

The first action was to conduct the requirements analysis of the concept.

A survey of existing efforts (national as well as international) was initiated to give answers to these questions. The CERMA (Centre d'Etude et de Recherché en Medecine Aerospatiale) tackled the problem of operational expertise. Extensive work with Mirage F1 reconnaissance pilots provided a lot of clues resulting in the identification of various pilot behaviors correlated with pilot profiles. These studies were conducted on pilot's cognitive activities during high speed, low altitude penetration missions over a four year period. During these missions, expert pilots and

novices showed different cognitive processes. Pilots' strategies depend on their knowledge of their own competency. It was also noticed that priority is given to short term activities like flight control. Medium and long term activities, like navigation and tactical planning, are only engaged in when the flight is stabilized. The flight plan is never executed as defined during ground preparation. These studies highlighted how the pilot's metaknowledge influences mission-preparation as well as his instant reactions. A joint activity with Dassault allowed the mapping of Pilots intuitive aspirations for new assistance into the reality of Mirage 2000 and Rafale advanced designs. This established the feasibility of adaptive aiding for the Man/Machine relationship concept. Building on this requirements analysis, Dassault oriented the design of the Copilote Electronique toward a Pilot-in-the-loop adaptive aiding system.

In parallel with these studies, the French Defense Services asked Dassault to carry out a feasibility study in order to validate the technological KBS orientation of the project.

This action followed previous attempts in the field of dogfight simulation which had shown that Pilot's experiences could be modeled in the form of rules in a reasoning system in order to fly a combat aircraft against a single opponent. A software environment to perfect the expertise in multi-aircraft combat tactics by automatic learning was created later with French Defense Services support. One difficulty encountered in this first attempt was the lack of an adequate development infrastructure. Only a homemade version of Lisp could be used on an IBM mainframe resulting in poor performances. A new project prototyping was therefore realized with more mature software environment (ART shell and Common Lisp on a Symbolics machine). The resulting mock-up demonstrated that KBS technology could work on a tactical advice subset of the Copilote Electronique. It included some tactical assessment, planning and plan selection. The four expert systems which were developed used 400 ART rules and 3000 objects. Real time performance was not reached, but ways of improvement could be foreseen. This work drove the architectural design of the Copilote Electronique toward a multi-agent distributed AI approach. It revealed the difficulty of mastering multi-expert architectures and identified the necessity of specific methods and tools (other than rapid prototyping ones) for the full scale development.

At this stage both the ergonomic concept and the technical orientation were considered feasible.

6.3.1.3 Iterating on Previous Experiences : the Technical Studies Phase

At the end of the first iteration of the Copilote Electronique program, three main unanswered questions were identified:

- Is it possible to capture enough expertise to create real assistance for pilot reasoning?
- Is the KBS technology mature enough for a real development?
- Is the Aerospace community able to integrate such a new concept in a current avionic system design?

Two types of studies were then initiated to address these questions. A first set was dedicated to prototyping of various expert functionalitys. A second set considered the elaboration of proper tools to populate a program development infrastructure.

As the environment of a military aircraft is not a static, well known and precise one, a research task addressing the management of uncertain and temporal information appeared necessary. This has been undertaken by Dassault in collaboration with LIFIA (a research laboratory in Grenoble), under DRET contract. The approach was to take advantage of the operational expertise gained from heuristically handling the quality of information. This allowed reasoning about the certainty of information, the quality and interest of hypothetical worlds, and the confirmation of information during the mission.

Another expert system prototype was developed by Dassault for rapid diagnosis of Mirage F1 equipment on return from a mission. It has been mocked-up under STTE (a French Defense Technical Service) sponsorship and in collaboration with Dassault Electronique. It performed as a realistic filtering module for the RAFALE A demonstrator aircraft. The mock-up was implemented with EMICAT, an expert system shell on top of Prolog. This work gave some initial clues about addressing real-time issues of KBS onboard implementations.

At the same time many aerospace companies were conducting preliminary prototyping of expert systems in the G&C domain.

For instance, SAGEM, as an Inertial Navigation System (INS) manufacturer, and the French agency STTE decided to launch SEAN - an Expert System For Navigation Assistance Aboard Fighter Aircraft. This project, started at the end of 1986 and ended in 1989, can provide a view of the community background at that time.

Increasing precision of navigation equipment, as well as on-board computers performance and diversified means of navigation updating have improved navigation performances but has also complicated the pilot's tasks.

Two main goals were assigned to the expert system :

- assist the end-user in performing critical choices (such as validating a navigation update, selecting the best out of 6 navigation choices, selecting the best inertial unit, ...) ;

- assist the end-user in understanding and qualifying the supervised system (such as qualifying the optimal navigation, diagnosing the present navigation update, diagnosing the inertial units, ...).

In order to fulfill these requirements, it was obvious that one had to work with several experts :

- flight test pilots and co-pilots (end-users) - in order to identify their needs and requirements ;

- inertial systems designers ;

- flight test engineers involved in tuning and repairing the INS ;

- experts in Kalman filtering.

The methodology and project management were based on the rapid prototyping approach. As explained in chapter 5, a KBS development differs considerably from a conventional software development. In this project, two methodologies were selected for use. First, among specification and design software methods, the KOD method was selected for knowledge acquisition.

Secondly, an upstream knowledge validation was conducted which was based on simulations. In this manner, the acquired knowledge was validated as soon as possible in the application life cycle. After completion, a validation of the entire demonstrator was performed.

A demonstrator was implemented on a Symbolics machine using the expert system development shell ART. This software was coupled with a complete simulation of the INS. It showed the improvements offered by an expert system. Implemented knowledge performed the diagnosis of six of the most common inertial error sources and supervised navigation updates. The project also revealed the limitation of the current knowledge based systems supporting environment (cf Chapter 5).

Of course many similar developments were conducted in parallel during this first iteration by other aerospace companies with useful results for the orientation of following phases of the project. In particular, many KBS development tools were used and evaluated.

Dassault conducted technical studies, mainly driven by the distributed AI perspective. In the context of the Copilote Electronique program, it appeared necessary to define how the industry partners would interact with Air Force operational experts for knowledge capture and refinement.

The CERMA studied a specific methodology for eliciting and formalizing pilot's expert knowledge. This methodology will be used for the Copilote Electronique Exploratory Development. It covers various aspects of expertise elicitation, such as:

- session preparation,
- role of the session leader,
- interview technique,
- formalization technique including :
  - full text,
  - object descriptions,
  - test descriptions,
  - action descriptions,
  - strategy descriptions,
  - mission descriptions.

This methodology was derived from previous CERMA work with Mirage F1 pilots and has been tested by Dassault while eliciting Mirage 2000 Pilot expertise. The development concerned an intent recognition module. It was implemented in Common Lisp and Flavors with an actors mechanism. This implementation was done in order to test the methodologies proposed for the full Copilote Electronique project. In particular the method and tool for capturing pilot's expertise were tuned during this study. This gave useful insight about the knowledge acquisition process.

Another important activity was initiated to survey the real time performance of AI mechanisms. This performance must be forecast regarding the constraints inherent in the suitability of an expert system for airborne use. For the Copilote Electronique project, realistically hypothesized core avionics anticipated for the year 2000 are used. With various partners, Dassault explored several potential solutions such as parallel processors, symbolic processors, risc processors, etc. For instance, a parallel implementation of the initial mock-up has been realized. It results in a network of 25 tasks functionally equivalent to the four expert systems. The performance improvement achieved, as well as the re-engineering time observed, confirmed the feasibility of an iterative optimization after the initial functional design.

Similar work was performed by Dassault with ONERA on the Multi-agent problematic aspects of the Copilote Electronique. The architectural work seemed very tedious and not easy to project to future real-time implementations. Thus tools to help the architect appeared necessary. This work resulted in the development of a design tool called SAHARA. SAHARA is a French acronym for "Simulateur d'Architecture Hétérogène, d'Agent et de Ressource Active" (Heterogeneous Architecture, Agent and Active Resource Simulator). SAHARA's main goal is to help to design a multi-expert system in vue of future producability. It is assumed that such a system has to interact with the outside, is made of several agents, (i.e. expert systems reasoning simultaneously and interacting asynchronously) and has to take temporal constraints and limited resources into account. SAHARA helps the designer face the multiplicity of choices and complexity of interactions by helping him describe the architecture and evaluate its performance.

The modeled architecture consists of three levels; functional, structural and virtual resources. The functional description is given in terms of a hierarchy of functions (as in SADT methodology) where the inputs, outputs and behavior of each function, and the data flows between functions are specified. SAHARA helps the designer determine if the implemented function corresponds to the desired function. Structural description includes the definition of implemented procedures (size, time response, exchanged data with other procedure...), used memories and communication protocols between procedures. SAHARA simulates the procedure behavior in terms of processing durations and memory allocations. Resource description includes the definition of available computing facilities, available memory and communication facilities (flow rates, access time...). SAHARA simulates the described architecture with chosen scenarios and supplies information on measures of performance. All these measures help the designer evaluate his architecture and, by doing so, improve it.

These technical studies gave the aerospace community a strong background to investigate the cognitive aspects related to the various areas of expertise associated with decision aids, and the computer science aspects relative to the implementation of artificial intelligence techniques in real time airborne systems. They were conducted in order to better plan the future development of the Copilote Electronique.

6.3.1.4 Planning For Development : The French Federative Project Definition Phase

According to these first results, it appeared necessary to initiate a multi-expert, multi-industrial federative project.

Asked by the French services to federate these competencies via a major national program, Dassault identified the major industrial stake holders and initiated discussions with them in order to take advantage of their past experiences. It happened that these experiences were very diverse and adequately covered the multiple aspects of the KBS problems, with each project bringing an interesting viewpoint.

The preliminary phases then shifted to a more precise project definition with planning and risk analysis. The first constraint on the plan definition was to respect the proper vision of the concept. To satisfy that constraint it was essential to identify a general architecture properly backed by fundamental considerations (acceptable by any stake holders) and directed towards the final project consideration, i.e. the challenge of considering the human in the design and maintaining enough flexibility to resist the economical and industrial tendencies during the program execution. Dassault proposed such an architecture at the level of the main assistance domains and worked with industrial and official partners to refine this architecture in a top down design approach.

This process turned out to be difficult and risky. Difficult because the concept was still fuzzy and with each company having a broad scope of competencies, the limit of each companies assistance domain was controversial. Risky because each partner had its own goals, culture, and methods. Consequently, the system design could have resulted in a collection of nice, but incoherent, functionalities.

Several actions were initiated to reduce this difficulty. An international cooperation was initiated by Dassault with Lockheed in order to augment the French approach with the US team experience and thereby avoid several drawbacks during the program planning. Initial experiences in the development of the Copilote Electronique, as well as the technical exchanges with Lockheed Pilot's associate team, led to the fact that conventional knowledge engineering techniques using questionnaires and interviews are not sufficient to provide implementable and secured knowledge for Pilot aids. This confirmed the view that knowledge engineering techniques, investigated in order to tackle the problem of such a complex AI system development, are interesting in the perspective of large MMI designs in aerospace systems. Those techniques should be used for capturing expertise in the initial design. They should then be supplemented by extensive knowledge evaluation and correction in simulation and, in some cases, backed by automatic knowledge generation tools.

Collegial work was performed to bring the partners to a consensus on the project shape. Lots of work was initiated to refine the initial "pyramid" architecture (figure 6.24) into lower levels of functional descriptions resulting in SADT-like formalization of each module of the architecture. The use of SAHARA at this stage to confirm the architectural choices and optimize the producability of the future system is an important part of the approach.

Another lesson from past technical studies concerned the difficulties linked with the choice of a generic development shell for all of the assistance domains. The large number of KBS environments tested showed the desire to keep flexibility and optimized tool selection for each domain. Consequently, the development phases are planned upon an open software development architecture (ranging from languages like LISP, C and ADA, through specialized languages for constraint programming like CHIP or knowledge based operating systems like KOS, to complete development shells like MUSE, ART, EMICAT or ILOG-RULES) controlled through a set of adapted communication and cooperation protocols. This of course requires a precise definition of a common semantic at the exchange language level. As a result of the initial iterations, a technical principle was established to facilitate these exchanges via the intent planning paradigm. In the open software development environment, each domain expert should be tuned prior to optimizing the overall real time performance of the Copilote Electronique.

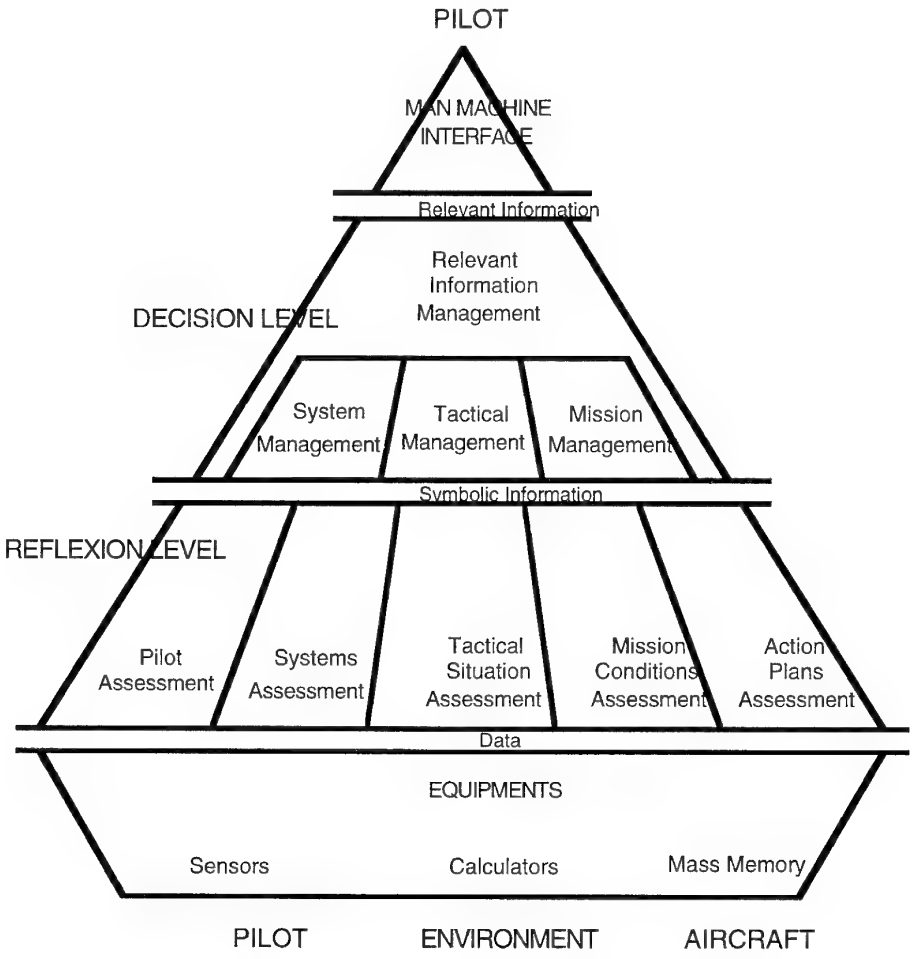


Figure 6.24 Copilote Electronique Architecture.

This phase ended in a seminar where technical contributors and economic partners gathered to express a common set of objectives, a coherent view of the concept and an analysis of the risk associated with the project plan definition. At this stage a complete development phase was proposed to the French official services and the Copilote Electronique program entered another part of it's life-cycle.

6.3.1.5 Concluding Remarks

The Copilote Electronique project will now proceed in two stages in order to integrate an adaptive assistance function in the aircraft. The KBS will first be used in a simulation context providing an operational demonstration without jeopardizing the aircraft safety before an on-board system implementation. Then a real-time prototype will be tuned to meet the constraints of guidance and control systems. The management plan of these two stages is the result of the lessons learned from previous cycles of this project.

A first lesson of the preliminary developments is that the software infrastructure should be very flexible and should ease the tuning of knowledge bases rather than insuring real time performance first. Future development infrastructure will accommodate the tools best suited for each planning domain and will leave an opportunity for new partners to join the team.

Another important lesson is that a distributed architecture should be continuously tuned and mastered through early simulation tools. This is necessary to prepare the next phases of the life cycle in the iterative life cycle perspective. One should deal with the interest of the architectural choices (i. e. the physical distribution, the communication protocols or the real time properties allocated to various modules) at the earliest stage of the development, long before the real code can be observed.

Finally, the experience with the specific methodologies in handling the complexity of multiple domain, interactive agents was very helpful. The metrics observed during the initial iterations are being used to extrapolate the cost of the knowledge acquisition phases of future developments. Specific tools are designed to help the knowledge representation and, more specifically, to insure the consistency of expertise throughout the various domains and the various partners.

It can be stated that the iterative process followed by the Copilote Electronique project during the preliminary phases is a fruitful experience in view of the coming development. It is difficult at this time to determine the cost reduction effect of this approach, but one can argue that such a complex program cannot avoid such a risk reduction approach.

Careful management of a large KBS life cycle is an important product of a project initial cycles.

6.3.2. Flight Line Diagnostic Systems

The magnitude of the maintainability problem in currently fielded tactical aircraft can be found by examining the Re-Test OK (RTOK) and Can Not Duplicate (CND) percentages. RTOK percentages are representative of the number of unnecessary removals and CND percentages are representative of the number of times a failure indication cannot be diagnosed. Typical CND and RTOK figures for the F-16A in 1988 were 31% and 22%, respectively. As more sophisticated avionics are introduced, these figures, even in the presence of more sophisticated Built-In-Test (BIT), have continued to rise. For example, the Flight Control Computer RTOK rate for the F-16C over the period spanning January 1990 to September 1991 averaged approximately 40%.

Complex avionics systems are typically designed to meet a 95% BIT coverage requirement. The remaining 5% of undetected faults must be located and diagnosed by the human technician. Since that 5% often represents the failure modes which a skilled designer couldn't find a good way to detect, finding the problems represented by this final 5% is by definition a challenging task. Even for failure modes which are covered by BIT, the age and experience level of maintenance crews are dropping, exacerbating the problem of correct diagnosis. In a study by Rome Air Development Center (RADC), 65% of RTOK instances were caused by the technician failing to use the available information to best advantage [6.44]. As the United States Air Force pushes to reduce the number of maintenance specialties, and goes from three level (depot, intermediate and flight line) to two level (on-aircraft and off-aircraft) maintenance, this reduction in specific skills is likely to continue and worsen. Classical approaches to training for this expertise involves classroom instruction and simulator time, which is time-intensive, costly, and ineffective, if we are to judge by the RADC study. Once the technician emerges from classical training, he is given Technical Orders which describe:

- how to determine what is wrong (Fault Isolation Manual)
- how to perform procedures (Job Guide)
- how the aircraft is built (Illustrated Parts Breakdown)
- how the aircraft is wired (Wiring Diagrams)

Unfortunately, this represents a mass of paper which fills a good sized file cabinet. Navigating through this information is a formidable task, even when the information is used in a favorable environment. When carried to a flight line, using the information is difficult at best.

Worse yet, even this mass of paper is incomplete. For example, the Fault Isolation Manuals (FIMs) contain a statically determined fault tree, in which each node of the tree represents a decision based on information obtained from performing a test procedure. It is common when diagnosing a complex problem to see the phrase, "Refer to schematic" as the final instruction. This means that either the technician has made an error in diagnosis, or the problem was not considered likely enough by the authors of the Technical Orders to warrant explicit directions for remediation. In either case, it is a frustrating experience, and one not manageable by most maintenance technicians. "Refer to schematic" is commonly followed by, "Call the chief".

6.3.2.1 Approach: Discussion of Solution

6.3.2.1.1 Why KBS?

Classical AI texts, such as [6.45] state that KBS (actually, expert system) approaches are suitable when the development task is "possible, justified and appropriate." This common-sense statement is developed in terms of a number of attributes which the aircraft avionics maintenance task clearly has in abundance: the task is cognitive, there are experts, the experts agree on how to solve the problem, and the task is "do-able". In addition, correct performance of the task is important in terms of readiness and cost savings, and human expertise is being lost as experienced crew chiefs retire from the Air Force ranks.

6.3.2.1.2 Benefits of KBS Technologies for this Application

A KBS system has the potential to significantly reduce costs associated with aircraft maintenance in a number of areas. Some of these benefits are indirect (reduced training, improved maintenance data collection and distribution, better morale), while others are more directly attributable and measurable (reduced maintenance man-hours per flight hour and reduced spares requirements).

Knowledge-based systems are usually made up of a repository containing the knowledge about a specific field and "reasoning" modules that uses knowledge within the repository to solve a specified problem. These kinds of systems are also able to provide explanations about why and how a solution is found. Moreover, changes within the domain don't imply rebuilding the whole system, but rather simply updating the knowledge base repository.

6.3.2.2 Implemented Systems

Though it is evident that KBS has clear benefits to offer, what *kind* of KBS to implement is still an open question. (See section 4.4 for a further discussion of alternative approaches). We present here two examples of successful KBSs which aim to solve the same set of issues. The first, Aircraft Diagnostics and Maintenance (ADAM), uses a symptom-to-fault mapping (rule-based) approach. The second, Flight Control Maintenance Diagnostic System (FCMDS), uses a functional connectivity model (model-based) approach.

Nonetheless, the two systems, since they solve a common problem, have a number of common attributes. Both

- put the user in charge of selection of the next test to perform (though the diagnostic system suggests appropriate tests)
- have extensive graphical and other help and explanation capabilities
- record the diagnostic session, for later review and potential training opportunities
- separate the knowledge from the inference procedure, allowing re-use of the tool for other applications
- provide a means for completeness and consistency checks of the knowledge base
- went through a process of initial prototyping, field test, and final implementation.

6.3.2.2.1 Aircraft Diagnostics And Maintenance (ADAM) - A Symptom to Fault Mapping Approach

ADAM [6.46] is composed of a knowledge acquisition module and a run-time module. The first allows domain knowledge to be directly captured by having the expert "draw" his knowledge using a graphical tool, making knowledge acquisition and updating simple and rapid. The run-time module provides all the capabilities for the

consulting session. The user indicates the symptoms and ADAM supports him to find the fault that caused such symptoms.

The troubleshooting steps sequence is not rigid: when more than one action is possible, the user can choose which one to execute first. ADAM flexibility is further enhanced by features that allow the user to define the components replacement order. ADAM also performs search optimization, eliminating "unnecessary tests", i.e. those tests that have lost their power to differentiate between different search paths due to knowledge already obtained during this diagnostic session.

A version of ADAM has been implemented by Alenia for the entire AMX aircraft, including:

- Airborne auxiliary power
- Air conditioning
- Anti-ice
- Canopy control and indicating
- Crew escape
- Electrical power
- Engine
- Fire protection
- Flight control
    - Aileron
    - Electronic flight control
    - Elevator
    - Lift augmenting
    - Rudder
    - Spoiler
    - Stabilizer
- Fuel
    - Air to air refueling
    - Defueling
    - Distribution
    - Indicating
    - Refueling
    - Transfer
- Hydraulic generation
- Indicating and recording
- Landing gear
    - Arresting hook
    - Extension and retraction
    - Ground/flight safety
    - Ground operation
    - Position and warning

- Steering
- Wheels and brakes

- Lights
- Oxygen
- Rain dispersal
- Weapons
  - Air to air missile
  - Gunnery
  - Stores management
- Avionics
  - Communication
  - Navigation

6.3.2.2.1.1 Knowledge acquisition methodology

Two kinds of knowledge sources are generally available: the knowledge derived by operational experience and design data. To obtain the optimal fault search paths, one must take into account both of them, since during the knowledge acquisition process, the expert has to reason using design knowledge and to verify the resulting paths by using his operational expertise.

In order to elicit the knowledge in a systematic way, a knowledge acquisition methodology consisting of the following steps has been adopted:

1) Analysis of an aircraft system or subsystem and collection of related documentation.

2) Definition of the components list (e.g. LRU's) that are diagnostically meaningful.

3) Definition, for each component, of all possible fault modes. This stage allows us to discover systematically all the faults of the aircraft.

4) Definition, for each fault mode, of all failures (malfunctions or symptoms) caused by the fault itself. The resulting set of cause-effect chains

   Fault ---> Malfunction [---> Malfunction ...]---> Symptoms

   is called a Fault Tree. This stage systematically defines all the possible symptoms which can be observed on the aircraft.

5) Definition, for each symptom (or set of symptoms), of the optimal diagnostic path, i.e., the best tests sequence leading to the isolation of the fault. The result of this step is a set of chains

   Symptom(s) ---> Test [---> Test ...] ---> Faults

   called the Decision Tree that represents the expert's reasoning.

Through steps 4 and 5 the knowledge engineer can perform a cross-check between fault tree and decision tree, verifying that all faults causing a symptom are the leaves of the decision tree having that symptom as root. This check also ensures that the knowledge base is complete with respect to the identified symptoms. This methodology has the added benefit that the process matches the domain expert's model of reasoning quite well, making use of the system natural and comfortable.

6.3.2.2.1.2 Architecture

The system, shown in Figure 6.25, has two main modules: the knowledge acquisition module and the run-time module. The knowledge acquisition module provides an easy and "user-friendly" mechanism to update the knowledge

base. The run-time module supports the user during diagnostic sessions. Both modules use the decision trees as a means to represent the system knowledge. There are two more modules that manage the fault rate and the images and connect the system to the respective databases.

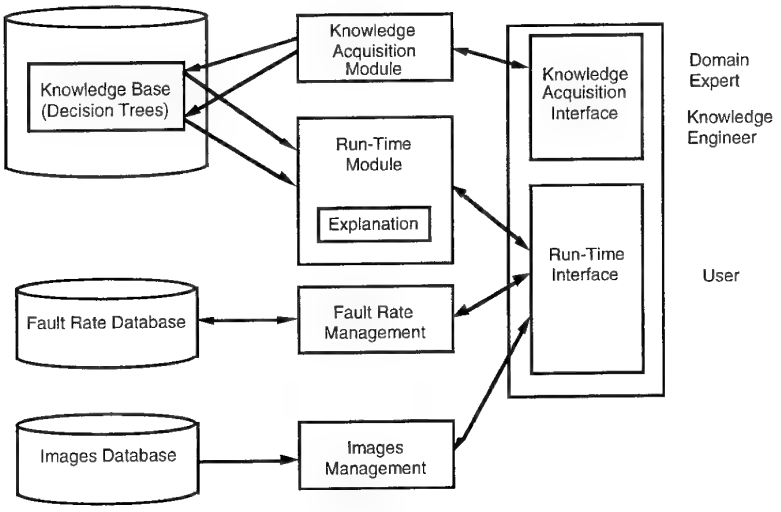


Figure 6.25 ADAM Architecture

Knowledge Base

The knowledge base is made of several parts, each representing a different system or subsystem of the aircraft. The knowledge is set up as decision trees: the sequence of steps that starts from the reported symptoms and leads to the fault identification, through a series of tests.

The internal representation of the decision trees is set up as objects linked via slot values. The following types of objects are used:

- Symptoms: roots of the decision trees.
- Tests: intermediate nodes of the decision trees; each test has two or more answers (typically yes or no) that determine the links to other objects (either other tests or faults).
- Faults: leaves of the decision trees.

These objects are linked via values stored in slots. For instance, a symptom is linked to tests or faults through the value of the "Hypothesis" slot.

Decision trees can share partial common paths. The complete set of decision trees makes up the decision net. There is a decision net (i.e., a knowledge base) for each subsystem of the aircraft.

Run-time Module

Fault detection can be carried out at two different levels: in the hangar and on the flight line. The most complete and complex case is hangar troubleshooting, where all test equipment and documentation are available: the diagnostic process corresponds exactly to decision tree exploration. Fault detection is done by means of a module that explores the decision net, starting from the detected symptoms. The module displays the first test of the diagnostic path, the user performs it, then indicates the result to the system. The module uses this result to continue net exploration. When the system finds a fault, it asks the user to fix it and to give a confirmation about the failure rectification. The process goes on until fault (i.e., a leaf of the decision tree) rectification is confirmed. The search algorithm explores the net using the typical features of object oriented programming (e.g., message passing).

In flight line troubleshooting, the system proposes a subset of all possible causes: this subset is composed by all faults that can be repaired at the flight line, i.e., in an easy and fast way and with the simple and portable test equipment available there. Typical faults repairable at the flight line are LRU malfunctions, circuit breakers not plugged in, etc. These hypotheses are suggested as alternatives, so that the user can choose which one to consider first.

In both cases, the system provides several features for explanation:

1) graphical display of the proposed diagnostic path (decision tree); so, at the beginning of the session, the user can know what the system will display to him and which are the candidate faults causing the initial symptoms;

2) graphical and textual display of steps (tests, substitutions) already done;

3) local fault rate (based on the fault reachable from the current step) and global fault rate (based on all the faults of the current system or subsystem);

4) display of significant images representing block diagrams or manual drawings.

The system is able to record the complete diagnostic path followed by the user. This capability can be used to pause and store the diagnostic session, so that ADAM can be immediately used for a new session. Later, the user can resume the interrupted session (e.g., after the execution of a complex and time-consuming test). Results of the session are also used to update the fault rate database, resulting in improved performance for future diagnostic sessions. In normal operation, ADAM suggests the test which the domain expert considers optimal. The user may choose to follow another path through the decision net, or directly postulate a faulty component. When the user exercises this override capability, ADAM automatically tracks what portions (nodes) in the decision tree no longer provide diagnostic information, and eliminates them.

This procedure elimination is shown in figure 6.26. Suppose the user, instead of verifying the functionality as proposed by the first test, directly selects the fault "IFF TRANSPONDER SB-X001". The system highlights the chosen fault (shown on the left side of the figure). If the fault is not rectified, the functionality test becomes redundant, since after the substitution of "IFF TRANSPONDER SB-X001", the functionality of the component is almost certainly correct. So, the system will mark this test as useless (colored in dark gray in the right side of the figure), bypass it and will propose the next continuity test (colored in light gray). In this way, the system is able to optimize the diagnostic search.

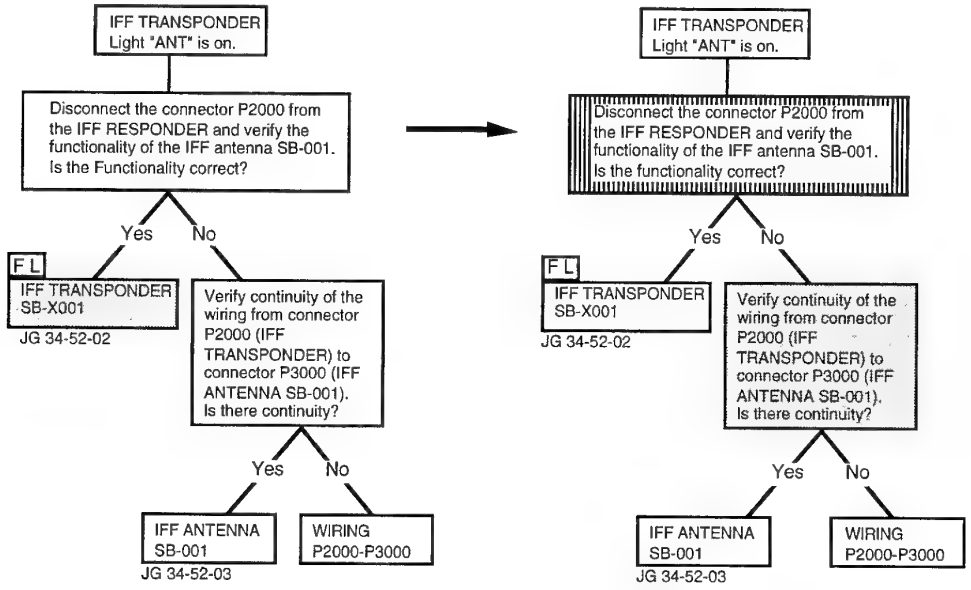


Figure 6.26 ADAM Automated Procedure Bypass

Knowledge Acquisition Module

The knowledge acquisition module allows us to represent the domain expert's reasoning in a very natural way using the decision tree formalism. The user simply "draws" the trees using the system facilities; the module directly translates this graphical representation into the system's internal formalism. To create or to modify the knowledge base, the user doesn't need to learn either Artificial Intelligence or programming techniques. This results in rapid and painless knowledge acquisition. The knowledge acquisition module also provides a capability to link additional information to the decision tree nodes: references to Job Guide manuals, drawings, images for explanation, and so forth.

A knowledge documentation function prints the current state of all decision trees. Thus, it is very easy to obtain documentation about the whole knowledge base, that can be used, for example, during the knowledge base validation stage.

6.3.2.2.1.3. Software/Hardware Platform

The current system is built using the Goldworks III toolset that provides a LISP programming environment. It runs on the Windows 3.0/3.1 environment on a 386 P.C. and is integrated with images drawn with standard products running in the Windows environment or input via an optical scanner.

6.3.2.2.1.4. Future Directions

The next version of ADAM will make use of different tools. It will be written using C++, in order to solve problems of portability, data persistence and performance. Data and knowledge will be stored in an object oriented database. Further, we plan to incorporate a "deep" module, able to find the faults by reasoning on the structural and functional model of the aircraft systems. The reasoning on the deep model could also be integrated with the one on the shallow model created so far (model reasoning integration).

6.3.2.2.2. Flight Control Maintenance Diagnostics System (FCMDS) - A Functional Connectivity Approach

For more than seven years, Honeywell has been developing a system, the Flight Control Maintenance Diagnostic System, to perform diagnostics and act as a maintenance aid to F-16 flight line technicians [6.47]. The system was developed under contract to the Flight Dynamics Laboratory of Wright Laboratory, contract number F33615-85-C-3613.

6.3.2.2.2.1. Technical Approach

FCMDS is an end-to-end system with support for pilot debrief, capture of logistics information, automated flight control system status debrief via the 1553B bus, diagnostics, electronic maintenance aiding, and electronic maintenance action report generation. At the core is a functional connectivity model which expresses the dependence of a given component on the contributions of its connected neighbors. Components are represented down to the "sub-LRU" level, that is, functions within a Line Replaceable Unit. The static fault tree captured in the Fault Isolation Manual has been "flattened", so that any procedure (e.g., measuring and reporting a voltage between two pins) can be executed at any point in the diagnostic session. In many cases, diagnostic procedures have been added to help identify the faulty component in an ambiguous set. Most procedures take the form of stimulus-response actions: the technician applies a stimulus to some point in the system, and observes the response. If the response matches (within some specified tolerance) the expected response, the "goodness value" of all elements in the functional path just tested are credited. If on the other hand, the response is flawed, all components in the functional path are debited. Overlapping paths are tested to apply different credit/debit transactions. When a clear faulty component emerges from this process, a remove and replace action is recommended.

6.3.2.2.2.2. Maturity of Solution (Process Description)

As with most programs with this sort of longevity, the nature of the issues addressed, and the work performed, has changed over the life of the program. The system began as a research prototype, but over time has evolved into a workable approach that has undergone first preliminary field testing, and subsequent more exhaustive field tests as reported below. These changes have been driven by a number of factors, including: 1) technical advancement both within and outside of the contract scope, 2) changes in the perceived need, 3) the identification of previously unknown constraints either in the user environment or inherent in the then contemplated approach.

The evolution of the FCMDS effort is presented, starting from the "humble beginnings", in which it was unclear what the correct approach would be. The pieces of the solution which together comprise a workable approach to such a system are discussed in turn: 1) diagnostics, 2) hardware, 3) user interface, 4) languages, and 5) development tools. For each piece, then, we discuss the influences and progress made to date. We discuss the future that we see as appropriate for each element, and end with some conclusions and recommendations in the hope that others who are at varying stages in this all-too-familiar evolution can benefit if not from the specific recommendations, then from the exercise of examining the probable course of events on their project. (Chapter 5 of this report contains an extensive examination of the development process and environments).

Diagnostics

When the FCMDS effort began, in mid-1985, there were no commercial systems that served as maintenance diagnostic aids. Several demonstration prototype systems were available as guideposts. Unfortunately, there was no consensus on the "correct" approach to building such a system. The initial prototypic diagnostic system that was created for the FCMDS program used a qualitative model-based approach. Tests results were evaluated against the

model by tracing from an output measurement back to known input signals. If a test passed, the goodness measure of the components that were in the tested path was increased. If a test failed, the goodness measure of the components that didn't match the expected qualitative values was decreased. If a new test was needed to reduce the ambiguity group of the most suspected components, the test that best divided the ambiguity group in half was selected.

At the end of a six month development period, the first demonstration system was created. The diagnostic system detected faults in the pitch trim subsystem of the F-16A flight control system (FCS). This subsystem represented approximately 2% of the total FCS. The diagnostics worked extremely well. All postulated failures were isolated to the correct LRU.

Over the course of the next three years, the diagnostic system was expanded from 2% of the FCS to the entire FCS. One of the first casualties of the expansion was the level of detail in the qualitative representation. There were several problems: processing the large qualitative model required significant time, mapping quantitative to qualitative values and back was difficult, and verifying that the model represented the real-world vehicle behavior was extremely difficult.

The decision was made to represent the functional information flow of the system without the specific values. The Sub-LRU now held functional information of the form "Output signal X is a function of input signals Y and Z." Using this representation, automatic generation of diagnostic models from a CAD database is possible. (This is discussed further in section 4.1).

Also, as the model became large, the components that could have caused a test's failure could no longer be identified quickly. The component dependency trace that was performed for each test as it executed began to take longer than the technician's perceived patience level, so the test traces were "pre-compiled".

In retrospect, given time, any reasonable approach can probably be as successful for a small, limited demonstration system as our approach was. At the time, of course, we felt that we just needed to fine tune the diagnostic algorithm and fill in the knowledge bases. Time and time again, the size of the knowledge bases threw obstacles in our path. Execution speed was frequently a problem. We were fortunate that we were able to slowly increase the system size. When an execution speed barrier would present itself, we had time to analyze and correct it. If our early success would have driven us to immediately expand to the full FCS, the combination of barriers may have been too much to overcome.

Hardware

Initial knowledge based system software development was performed using Symbolics 3600 workstations. These workstations were developed specifically for use with the LISP language, and have built-in hardware to assist the most common operations in the LISP language. Coupled with the Symbolics operating system, these machines provided a very powerful platform for prototype symbolic software development.

Demonstrations of the FCMDS software required access to similarly equipped Symbolics workstations at the demonstration site. An essential element in the successful development of a knowledge based system is the ability to obtain end user feedback early in the software development phase. Few operational Air Force Bases had access to this type of computing environment, requiring demonstrations to be held at the FCMDS software development facility or Wright Patterson Air Force Base.

Portability of the targeted hardware was emphasized as a key element in the selection of a next hardware platform. Conflicting requirements were that the majority of the already developed software must be easily transported and run efficiently on the new target hardware. The goal of portability favored a laptop computer as candidate system hardware. Though a desktop system choice was sub optimal from a portability perspective, particularly when contemplating use in a flight line environment, it was at the time the only available form factor of the requisite computing resource.

The next FCMDS port was to the Agilis microcomputer system, which provides ruggedized computer processing resources closely matched with the desktop microcomputer system previously adopted [6.46]. Future directions for the hardware platform might include (as proposed for a recent Army maintenance aid program) voice generation and recognition with a helmet mounted display for hands-free operation. Elimination of rotating storage media (hard drive, CD-ROM, etc.) which is susceptible to impact damage may be accomplished with RAM modules made possible by 16Mbit (or greater) DRAM technology.

User Interface

The choice of user interface was determined, in large measure, by the hardware platform on which we were to demonstrate work to date. The Symbolics workstation provided a set of sophisticated tools for "programming the user interface", and manipulating windows and presentations on those windows. When we ported to the PC

environment, a graphics toolkit was chosen that provided support across a wide variety of computer display devices. New graphics display devices are easily supported using this approach since the only software that needs to be changed is the specific graphics device drivers. Rehosting of the user interface software was then performed using the selected graphics toolkit.

Remote demonstrations using the new desktop portable hardware and rehosted user interface software allowed the technical team to obtain significant user interface recommendations from the maintenance technicians. One of the best recommendations received was to display the cable wiring runs in a graphical format. The graphical depiction of wire runs shows a signal path from its source to its destination. A skilled technician could spend up to twenty minutes using the Air Force Maintenance Manuals tracing this signal path, with a large potential for error. Using FCMDS this process only required a few seconds.

Note that the most important evaluator of the interface is the user. Users don't always agree on what they want to see in the user interface, but the points on which they agree or disagree should be taken very seriously. Users often have different ways of looking at information. (Chapter 3 of this report provides a framework for understanding and managing these factors). If the diagnostic system can present the material in different formats to different individuals, all the users will benefit.

Standards, either de facto or adopted, will promote ease of use and interoperation with other tools. Much of our new software is already using MS Windows, which while neither knowledge based nor particularly cutting edge, is well-known and usable. We've found that software that isn't comfortable won't get used, no matter how good the underlying algorithms are (or how good we think the interface is). Support for individual differences by means of user profiles, and use of FCMDS as a training tool, tracking progress over a technician's career are both research areas.

Languages

The development for this phase was performed in LISP, a language normally associated with "artificial intelligence" research, but more recently finding favor as a rapid prototyping language. In a LISP environment, the linking step is bypassed. Binary files are loaded directly into the environment, and each definition found in the binary file becomes available for use. The compilation step can also be bypassed, if desired, by directly "evaluating" a LISP form. This feature allows on-the-fly redefinition of functions in the working environment, which substantially reduces the implement-test-debug-implement cycle time.

While this makes LISP an excellent choice for prototyping, as we examine moving this approach to other environments, particularly embedded ones, it is clear that LISP will not be a commonly supported language. Ada or C are more frequent choices. Thus, we have implemented portions of the diagnostic software in both Ada and C. Each language has its advantages and its disadvantages. In the final analysis, it comes down to using the language that has the best development environment, the best compiler, and the best debugger.

Development Tools

Prototyping a knowledge based system requires the use of software development tools with layers beyond the implementation language. These layers save the prototyper from having to re-implement commonly used features (such as a frame language). Initially the FCMDS technical team utilized the KEE tool running on the Symbolics workstations as both a development and demonstration system. Later this system configuration was used only for its sophisticated software development environment.

Portable demonstrations executing on a platform other than the development environment require either a common software approach or a translation methodology. The approach used by the technical team was a combination of both. For example, to support the KEE knowledge bases a Common LISP frame system was developed for the demonstration environment. Automatic porting of KEE knowledge bases was then made possible through the development of a knowledge base translator utility. Transitioning software from the development to the demonstration environment now could be easily accomplished in a few hours.

New development tools are being created to support incremental compilation of the model from the captured form to the run-time form. Design knowledge is no longer be explicitly represented as frames in KEE. Instead the design information is entered into a Computer Aided Design (CAD) tool and then translated into KEE frames using CAD database interpretation utilities. This suggests that the original design data serves as an appropriate source for model data. In a research setting, we have demonstrated translation of a detailed model from Electronic Design Interchange Format (EDIF) to the FCMDS format, and minimization of a model from the overly detailed CAD form to one which contains only the information required by the set of available tests.

6.3.2.2.2.3. Technology Transfer and Further Applications

The core technology developed under FCMDS has been applied to a variety of diagnostic aids, including systems for torpedo manufacture acceptance test, inertial reference unit test, and most recently, the Central Maintenance Computer for the next Boeing commercial airliner, the 777. In addition, the Air Force has undertaken a program to institute two level maintenance for the F-16 called the F-16 Integrated Maintenance Information System (IMIS), in which FCMDS plays a key role.

6.3.2.3. Status: Evaluation of the KBS Solution

6.3.2.3.1. Evaluation Criteria

The goal of the FCMDS system is to enable less skilled technicians to perform maintenance diagnostic sessions on the F-16 flight control system quickly and accurately. Thus, the evaluation criteria in a field test, conducted at the Luke Air Force Base between September, 1990 and June, 1991, were:

- troubleshooting time by grade
- unnecessary removals of LRUs

Diagnostic troubleshooting time was limited to a maximum of 45 minutes to reduce technician frustration with "insoluble" problems.

6.3.2.3.2. Test Results

Table 6.1 lists the number of false pulls by technician category for each of the failure cases using the technician's Standard Troubleshooting Methodology (STM), which in some cases did not use Technical Orders (T.O.s). No false pulls were encountered for technician experiments using FCMDS. The false pull rate increases for failure cases 4 and 5, reflecting that the relative difficulty of these failures is higher than for the first cases.

Table 6.1 False Pull Counts (RTOK)

	STM RTOK Count			FCMDS RTOK Count		
Case	FLCS	Non-FLCS	Total	FLCS	Non-FLCS	Total
1	2	2	4	0	0	0
2	1	3	4	0	0	0
3	1	3	4	0	0	0
4	2	3	5	0	0	0
5	5	1	6	0	0	0
Total	**11**	**12**	**23**	**0**	**0**	**0**

Table 6.2 presents troubleshooting time statistics using the STM and FCMDS. As expected the variance in the FCMDS data is much more stable than the STM data. Thus, on average FCMDS can provide accurate diagnosis in less time than the STM even with a more diverse work force.

Table 6.2 FLCS Troubleshooting Times

	STM			FCMDS		
Case	Average	Maximum	Minimum	Average	Maximum	Minimum
1	28	45	7	23	25	21
2	32	45	8	22	30	16
3	25	45	6	18	20	17
4	41	45	29	27	33	22
5	32	45	17	23	37	15
Net Av.	**31.6**	**45**	**13.4**	**22.6**	**29**	**18.2**

6.3.2.3.3. Conclusion

We have presented two successful approaches to performing diagnostics on the flight line. Although they use different KBS techniques, they have many features in common. As KBS tools see more common application, tools such as ADAM and FCMDS, are likely candidates for the first widespread uses of KBS, since they have fewer processor, real-time and certification constraints than on-board systems.

6.4 References

[6.1] Harris, D.F.; Morrisette, J.A.: Single Pilot IFR Accident Data Analysis, NASA Contractor, Rep. 3650, 1982

[6.2] Wiener, E.L.: Error Exposure and Control in High Technology Aircraft, in: Artificial Intelligence in Cockpit Design, Orly, 1989

[6.3] Committee on Automation in Combat Aircraft, Air Force Studies Board, National Academy Press, 1982

[6.4] Kraiss, K.-F.: Fahrzeug- und Prozeßführung, Springer, Berlin, 1985

[6.5] Rumelhart, D.E.; McClelland, J.L.: Parallel Distributed Processing, MIT Press, Cambridge, 1986

[6.6] Grossberg, S.(Ed.): Neural Networks and Natural Intelligence, MIT Press, 1988

[6.7] Dudek, H.-L.: Wissensbasierte Pilotenunterstützung im Ein-Mann-Cockpit bei Instrumentenflug, Dissertation, UniBw München, 1990

[6.8] Rasmussen, J.: Skills, Rules and Knowledge; Signals, Signs and Symbols, and other Distinctions in Human Performance Models, IEEE-SMC-13, No. 3, 1983

[6.9] Rasmussen, J., Duncan, K., Leplat, J. (Edts.): New Technology and Human Error, John Wiley and Sons, 1987

[6.10] Onken, R.: Knowledge-Based Cockpit Assistant for IFR Operations, AGARD GCP Symposium, Sept. 1990

[6.11] Onken, R.: New Developments in Aerospace Guidance and Control: Knowledge-based Pilot Assistance, 12th IFAC Symposium, 1992

[6.12] Dudek, H.-L., Onken, R.: Cockpitassistenzsysteme in zukünftigen Luftverkehrsszenarien am Beispiel CASSY, DGLR- Jahrestagung, 1992

[6.13] Prevot T.; Onken R.; Dudek H.-L.: Knowledge-based Planning for Controlled Airspace Operation as Part of a Cockpit Assistant, 53rd AGARD Symposium of GCP, 1991

[6.14] Zadeh L.A.: Fuzzy Sets and Applications: Selected Papers, J. Wiley and Sons, 1987

[6.15] Ruckdeschel W.; Dudek H.-L.: Modellbasierte Fehlererkennung in der Flugzeugführung, VDI- Tagung "Menschliche Zuverlässigkeit", 1992

[6.16] Wittig, T., Onken, R.: Pilot Intent and Error Recognition as Part of a Knowledge-based Cockpit Assistant, AGARD GCP Symposium, Oct. 1992

[6.17] Dudek, H.-L.; Wittig, T.; Onken, R.: Speech Input/Output for Communication between pilot and pilot's assistant system, Military and Government Speech Tech, 1989.

[6.18] Ruckdeschel, W.; Onken, R.: Petrinetz-basierte Pilotenmodellierung, Fachausschuß "Petrinetze" der GI, Berlin, 1993

[6.19] Prevot T.; Onken R.: Knowledge-based Planning for Controlled Airspace Flight Operation as Part of a Cockpit Assistant, HMI/AI in Aerospace, Toulouse, 1993

[6.20] Gerlach, M.; Onken, R.: Speech Input/output as Interface Devices for Communication between Aircraft Pilots and the Pilot Assistant System "CASSY", Joint ESCA-IEEE-NATO/RSG 10 Workshop, 1993

[6.21] Hoshstrasser, B.; Skidmore, M.; "The Design of an Object-Oriented Intent Inferencer for Integrated Systems" 10th Annual IEEE/AIAA Digital Avionics Conference, Los Angeles, 1991

[6.22] Hoshstrasser, B., "Inferring Operator Intent for Aiding in Man-Machine Systems " 12th Annual Symposium on Automatic Control in Aerospace, Ottobrun, Germany, 1992

[6.23] Hammer, J.; Small, R; "An Intelligent Interface in an Associate System" in Rouse, W. Human /technology Interaction in Complex Systems (Vol. 7) . Greenwich, CT: JAI Press

[6.24] Edwards, G.; Geddes, N.; "Deriving a Domain-Independent Architecture for Associate Systems from Essential Elements of Associate Behavior", Proceedings DARPA Associate Systems Workshop, Fairfax, Va.' 1991

[6.25] Holmes, D.; Retelle, J.; "The Pilot's Associate: Exploiting the Intelligent Advantage, " Proceedings from AGARD Conference No. 474 on Knowledge Based Systems for Guidance and Control, Madrid, 1990

[6.26] Lizza, C.; " Pilot's Associate: A Perspective on Demonstration 2 " AIAA Computing in Aerospace, 1989

[6.27] "Meeting the Real Time Challenge: Pilot's Associate Program Phase 2, Demo 4 Technical Report Video," U.S. Advanced Research Projects Agency, Washington, D.C.

[6.28] Edwards, G. ; Hoffman, M.; Lafferty, l.; Shoop, J.; " A Real time Planner for Tactical Air Combat" AIAA Computing in Aerospace 8, Baltimore, 1991

[6.29] Voelckers, U., "COMPAS-RechnerunterstÅtzung fÅr die Anflugkontrolle", DGLR-Annual Convention, Hamburg, Germany, 1984

[6.30] Voelckers, U., "Dynamic Planning and Time Conflict Resolution in Air Traffic Control", in: Lecture Notes in Control and Information Sciences, H. Winter (Editor), Springer Verlag, Berlin, Heidelberg, New York, 1986

[6.31] Voelckers, U., "Computer Assisted Arrival Sequencing and Scheduling with the COMPAS-System", AIAA/ASCE International Air Transportation Conference, Dallas, 1986

[6.32] Voelckers, U.; Schenk, H-D., " Operational Field Test of the COMPAS-System at Frankfurt, Germany", AIAA Guidance, Navigation & Control Conference, Boston, Aug. 1989

[6.33] Voelckers, U., "Design Principles of Automation Aids for ATC Approach Control, Scientific Seminar of the DLR Institute of Flight Guidance, Braunschweig, Sept. 1990

[6.34] Muratore, J., Heindel, T., Murphy, T., Rasmussen, A., and McFarland, R., "Real-Time Data Acquisition at Mission Control," *Communications of the ACM*, Vol. 33, No. 12, pp. 19-31, 1990.

[6.34] Muratore, J., Heindel, T., Murphy, T., Rasmussen, A., and McFarland, R., "Real-Time Data Acquisition at Mission Control," *Communications of the ACM*, Vol. 33, No. 12, pp. 19-31, 1990.

[6.35] Aubin, Didier ; Megardon, P.; Parquet, Ch.; Brenot, JM.; Parrod, Y., "ARIANEXPERT : an expert assistant to assist in ARIANE's Post Flight Analysis", Launching Sites & Facilities to control spacecraft, PARIS May 1990.

[6.36] Brenot, JM.; Parod, Y.; Aubin, D.; Parquet, Ch., "ARIANEXPERT : Post Flight Exploitation of Ariane launcher telemetries recorded during the flight", FAIST'92, Fielded Applications of Intelligent Software Technologies,Toulouse-Labège, France, February 1992.

[6.37] Aubry, P.; Bracq,D.; Champigneux, G.; Gaudry, Y.; Havre, A., "Maquette de systeme expert embarquable sur avion d'armes". Colloque IHM et IA, TOULOUSE 88

[6.38] Morillon, D.; Conter, T.; de Cremiers, M., "SEAN: an expert system for navigation assistance aboard fighter aircraft". Conference IA et Defense AVIGNON 88

[6.39] Champigneux, G.; Gaudry, Y.; Gilles, A.; Nory, P.; d'Orso, M., "Vers une Intelligence Artificielle embarquee pour l'assistance au pilote: le Copilote Electronique". Conference IA et Defense, AVIGNON 89

[6.40] Amalberti, A.; Deblon, F.; Guengant, A., "Advances in Navigation Support Systems Based on Operational Pilot's Heuristics". AGARD, LISBON 89

[6.41] Barat, M.; Benhamou, Ph.; Ferber, J.; Nory, P., "Design and Analysis of Multi-agent systems with SAHARA simulation tool". COGNITIVA, MADRID 90

[6.42] Gilles, A.; Lebiannic, Y.; Montet, P.; Heudin, J. C., "A parallel multi-expert architecture for the Copilote Electronique". Conference IA et Defense, AVIGNON 91

[6.43] Salle, E.; Champigneux, G.; Reichert, O.; Gaudry, Y., "Development Methodology for Knowledge Based Systems Used in Guidance and Control. Application to the Copilote Electronique project". Conference IA et Defense, AVIGNON 93

[6.44] Rue, H.D. and Lorenz, R.O. "Study of the Causes of Unnecessary Removals of Avionic Equipment", RADC-TR-83-2, Rome Air Development Center, Griffiss Air Force Base, NY, 1983.

[6.45] Waterman, D.A., *A Guide to Expert Systems.* Reading, Massachusetts, Addison-Wesley, 1986.

[6.46] Baran, N. "The Ever-Shrinking Ever-Expanding Laptops." Byte, August 1989, pp. 90-96.

[6.46] Damiani, S., Ricci, F. and Valfre, E. "ADAM Aircraft Diagnostic and Maintenance", in Proceedings of the Third Conference of the Italian Association for Artificial Intelligence, Turin, Italy, October, 1993. (To appear).

[6.47] Bursch, P.M., Meisner, J.W., Jeppson, M. *Flight Control Maintenance Diagnostic System Final Report,* Defense Technical Information Center, Document number pending, January, 1993.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

In this report G&C functions in connection with aeronautical and space systems have been analyzed with respect to their potential for the application of Knowledge-Based Systems Technology. The **top-down** analysis of the different areas of application - aircraft, mission, air traffic control, spacecraft mission operations and life-cycle management - has shown large potential benefits from KBS. The results of this analysis are presented in Chapter 2. A **bottom-up** view of the same areas was carried out by investigating existing KBS examples. These results are presented in Chapter 6. A comparison of the top-down and the bottom-up views - together with a thorough consideration of the supporting technologies and scientific disciplines (Chapters 3-5) - has lead to some conclusions regarding the present status of KBS in Guidance and Control, their potential for further applications, the shortcomings of the supporting technologies, and the work to be done in the future. Many of these conclusions are presented in the main chapters of this report. The most significant ones are summarized in this chapter.

The example systems discussed in Chapter 6 illustrate the fact that **knowledge-based technology** is real and sufficiently mature for application. These examples stem from the areas of aeronautics, space and life cycle management. Some of the KBS representing *single functions* (e. g. diagnosis and planning) are already fielded and have demonstrated their high potential in the real environment. *Multi function* systems with more complex architectures are still in the laboratory environment, but testing and evaluation in simulated real-world conditions are underway. The first results indicate their high potential relative to conventional systems. In most of the examples considered, knowledge-based technology is applied to support, not replace, human operators.

Single function systems (see 6.1.4, 6.2.1, 6.2.2, 6.3.2), especially diagnosis and planning systems, find widespread application in aeronautical and space G&C systems, and experience from extensive field testing and from routine use is available. Ground-based applications (ATC, maintenance) seem to be easier to realize than airborne or space applications. This is primarily due to the existence of fewer processor (power, volume, weight, etc.), real-time operation and certification constraints for ground-based systems. Experience with fielded example systems indicates that, after successful implementation and acceptance by the human operator, there is a tendency to progress from single function to multi function systems. Supporting a single function of a human operator by knowledge-based technology in a multi function environment may lead to a shift of the bottleneck to another function in the functional chain. This tendency highlights the need for multi function knowledge-based systems with complex architectures.

Multifunction systems (see 6.1.1, 6.1.2, 6.3.1) are still mostly in the laboratory stage of development. Lessons learned to date from test and evaluation in simulated environments are as follow:

- Presently unresolved questions of validation and verification are hampering the use of KBS in safety critical applications. (see 6.1.3 for an example). KBS can only "assist" human operators and full automation is not possible in these cases.
- In order to be able be to "assist" an expert like a pilot, an air traffic controller or a space mission manager, a KBS must have comprehensive situation awareness and understanding, including the operator's intentions, which makes the system very complex from the beginning (See 6.1.1, 6.1.2)
- Architecture and development environment are important factors for the life cycle of KBS for G&C. They must be flexible enough to cope with changing situations and roles of the system in the total man-machine context (see 6.3.1)
- Human factors have to be considered from the beginning. The system architecture must also contain a path for degradation in case of failure of a machine function which the operator can not take over (see Chapter 3)

Human factors have a strong impact on the lay-out of KBS for G&C. This is especially true for multi function systems, because of the "assisting" role. The future role of automation will be to assist operators, not to substitute for them. This results in a human-machine partnership, where both partners must be informed concerning mutual status and intentions. Some basic principles have to be imposed:

- Automation should not be introduced for it's own sake, but to help the operator.
- Tasks should be allocated to human or machine according to what each does best.
- Attention has to be paid to the interaction and integration of man and machine.
- Machine execution of a task need not emulate the human reasoning processes.
- Users should be involved in the design and functional allocation from the outset

The following recommendations for the technical support of human cognition were derived from a detailed analysis of cognitive behavior (see 3.2):

- Improve information display and human-machine dialog.
- Support information management, decision making, problem solving and planning.
- Reduce educational and training requirements.
- Build error tolerance and error insensitivity into the system.

Optimizing the life cycle is the job of an overseeing function termed **Life Cycle Management**. There are three different ways in which KBS can be viewed in the context of life cycle management: a) The system whose life cycle we consider can be a KBS. b) KBS can be applied to some part, or phase, of the life cycle. (good examples are diagnosis systems for maintenance, a single function, 6.3.2). c) KBS can oversee the entire life cycle process (become, or aid, the overseeing function). In most cases this will be a multi function system (see 6.3.1).

Life cycle management is a very promising area for the application of KBSs because there are almost no processor, real-time or certification constraints. Some conclusions can be drawn from experience obtained with the examples considered:

- Often KBSs are the "best" possible solution among many other candidates. "Best" in this case means "easier to use by operators, cheaper, better performance, etc.", integrated over the life cycle.
- A KBS has the potential to significantly reduce costs associated with maintenance in a number of areas. Some of these benefits are indirect (reduced training, improved maintenance data collection and distribution, better morale), while others are more directly attributable and measurable (reduced maintenance man-hours per flight hour and reduced spares requirements).
- KBSs for life cycle management have complex architecture and functionality. This requires flexible software infrastructures and simulation tools for the tuning of the KBS to different phases of the life cycle. Specific tools for this application area must be developed.

Validation and verification is a key problem for KBS applications in safety critical G&C systems. This issue was not discussed in detail in this report (an example is discussed in 6.1.3), because it is the subject of other AGARD documents such as AGARD-AR-274.

Recommendations for **further G&C research** are:

- In parallel with many on-going and sometimes technology-driven single function applications, we must intensify the top-down and architectural considerations, in order to realize high-payoff KBSs for G&C.

- There are large areas in the G&C functional world where KBS technology has a great potential, but there is not enough investment in Research and Development (R&D). Examples include ; Multi-functional systems for battlefield management, for integrated mission and weapon systems management, and for air traffic management. An area with excellent prospects is life cycle management. Many of these areas are not hampered with extensive real-time and certification constraints.
- Although most of the enabling technologies are already sufficiently mature for application of KBS technology to G&C systems, they must be further developed, improved and tailored to the needs of aerospace engineers. This is especially true regarding the methods for knowledge acquisition and management.
- Full exploitation of the potential of KBS requires systems with complex architectures and full functionality. Development methodologies, shells and tools are needed. Standard software packages for the basic G&C subfunctions (monitoring, diagnosis, plan generation, etc.) must be made available.
- Development environments and tools for the design, testing, and evaluation of KBSs for G&C have to be developed so that complex solutions can be constructed from more basic functions. Finally, functional modules and the corresponding hardware have to be standardized.

APPENDIX A

APPLICATIONS OF NEURAL NETWORKS IN GUIDANCE AND CONTROL WORK SYSTEMS

A.1 Technology Overview

This appendix gives a brief overview of neural network (or connectionist systems) technology and a description how that technology can be applied to the monitoring, diagnosis, plan generation and plan execution sub-functions that have been defined and analyzed in Chapter 2 in the context of G&C related problems. Specifically, three applications are discussed: flight control (i.e., plan execution), image recognition and sensor fusion (i.e., monitoring and diagnosis) and route and mission planning. The discussion of the application of neural networks to flight control is an elaboration of that which has already been introduced in Section 4.5.1. For a more thorough introduction to the fundamentals of the technology of connectionist systems, see [A-1, and A-2].

A.2 Two Principal Classes of Networks

There are two principal types of neural networks: *feedforward multi-layer perceptrons* (MLP) and *fully connected recurrent networks* [A-2]. The MLP is a so-called static network that computes an output value given an input value and thereby defines a mapping between its inputs and outputs. The recurrent networks are dynamic in that they have an internal dynamic state, comprising state values for all of the nodes or elements within the network, that evolves in time. Typically, the "input" to a recurrent network is an initial value for this state and its output is the steady-state value to which the network settles over time (here we assume stable state dynamics).

The basic building block used in the construction of each of these two networks is the network element (or network node) illustrated in Figure A.1. As shown in the figure, the output of a given network element is a nonlinear function of a weighted sum of its inputs. This figure depicts the standard sigmoidal network output function nonlinearity. Modifications to the basic network element have been developed to overcome problems associated with so-called learning generalization (discussed later in this appendix). These modifications take the form of a variety of so-called localized nodes employing radial basis functions [A-2]. In particular, these modified networks have been applied in addressing flight control problems as discussed in Section A.5 below.

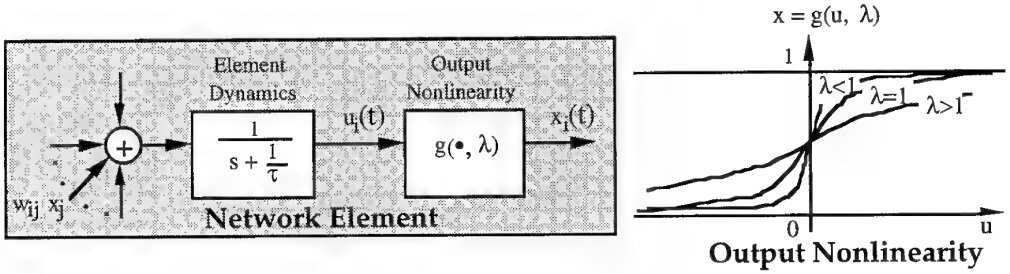


Figure A.1 Input-Output Relation for the i^{th} Neural Network Element
The inputs are the x_j (outputs of other network elements) and the weights are w_{ij}. The first order dynamics of the internal state u_i are present only for recurrent networks. The output nonlinearity is typically a sigmoidal like function as illustrated on the right.

A.2.1 Multi-Layer Perceptron (MLP) Networks

The principal function of a feedforward network is to approximate a desired mapping from its inputs to its outputs. If we let x represent the network input and y represent the network output, then the network provides an approximation $y_{net} = f_{net}(x)$ to a desired mapping $y = f(x)$. The approximation is "learned" by adjusting the input weights $w_{i,j}$ for each network element based on a sequence of training input-desired output pairs $\{x,y\}$. It has been shown that the class of feedforward sigmoidal networks is capable of learning arbitrarily complex continuous mappings [A-3].

A variety of approaches have been developed for adjusting the input weights for the network elements given a sequence of input-output training pairs. The most common (and simplest) form of learning (weight adjustment) is a gradient algorithm referred to as backpropagation. As with any gradient approach, step-size, local minima and either very flat of very steep cost surfaces can all result in slow or even improper learning. MLP learning problems typically exhibit all of these! There is an entire field of research dedicated to developing alternatives to the backpropagation search technique (again, see [A-2]).

A key issue that must be addressed in designing an MLP is defining the *network topology*. The network topology refers to the number of layers of nodes and the number of nodes at each layer (of course, the number of nodes at the input and output layers must be consistent with the dimensions of the input and output of the function to be approximated by the MLP). In addition, the topology is also characterized by the *connectivity* of the nodes in the network, i.e., which node's output provides an input for which other nodes. Figure A.2 depicts a three layer MLP, i.e., one that contains a single hidden layer. A variety of theoretical results relating the topology of the network, the difficulty or complexity of learning and the nature of the function to be learned have been developed. These results can be used as starting points in defining the topology of the network for a given application (see [A-2]).

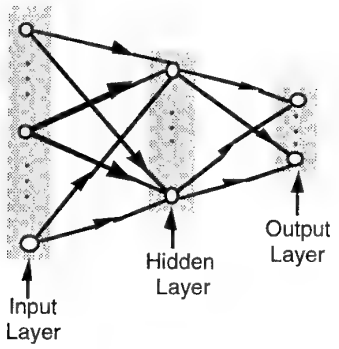


Figure A.2 A Three Layer Feedforward Multi-LayerNetwork.
In a feedforward network, the outputs of one layer (e.g., the input layer) provide the inputs to the next layer (e.g., the hidden layer)

A.2.2 Recurrent Networks

In contrast to feedforward sigmoidal networks, fully-connected *recurrent networks* (see the network topology in Figure A.3) have been applied in designing associative memories and in function optimization. For these applications, the network weights are pre-assigned and, therefore, no learning is necessary. More recently, learning approaches have been developed for recurrent networks. Combining learning with the dynamical system characteristic of recurrent networks presents the potential for learning models of nonlinear dynamic systems that can be employed in adaptive and sliding mode control applications.

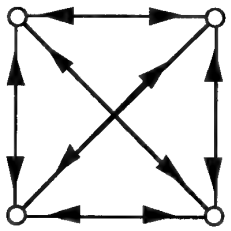


Figure A.3 A Simple Four Element Recurrent Network.
For the associative memory application, the network state (i.e., the values of the u_j in Figure A.1-1) is initialized to a *key*. For the optimization problem a random initial state is used. The network state settles out to a final value which for the associative memory problem represents the recalled value and for the optimization problem represents the solution to the optimization problem.

A.3 Issues and Limitations

A.3.1 Learning

Connectionist learning systems (neural networks) are appealing because (1) they are relatively simple in form and concept, (2) they can be used to learn and realize (i.e., approximate) general continuous nonlinear mappings and (3) they can be implemented in parallel computational hardware (both digital and analog parallel implementations have been developed). Error backpropagation [A-4] is a straightforward gradient descent algorithm that modifies (i.e., learns) MLP network weights incrementally to minimize a particular measure of learning error. The learning error is usually defined as the squared error between the actual network output for a given input and the desired network output. Since the network nodal functions are continuously differentiable, it is possible to calculate the gradient of the learning error with respect to the weights, and to then adjust the weights in the direction of the negative gradient. As with all gradient descent optimization techniques, there exists the possibility of converging to a non-optimal local minimum. Despite this, learning systems using back-propagation have been shown to find satisfactory solutions to a variety of problems including difficult, highly nonlinear control problems [A-5, A-6, A-7, A-8].

The problem of non-localized learning is often manifested in the form of input "fixation." Fixation can occur when the learning system input remains in the same small region of the input-space (near some point or trajectory) for an extended period of time. If learning is non-local, the mapping associated with some parts of the input-space can become unintentionally corrupted, as the learning system attempts to drive the learning error to zero in the neighborhood of the fixation point or trajectory, *without regard to the remainder of the mapping*. With respect to flight control, this situation could occur at any stable operating condition, since the principal states of the vehicle would be nearly constant. To combat the problems of non-localized learning in standard artificial neural networks, a number of corrective procedures have been applied, including batch learning, very slow learning rates, and non-contiguous input sequences. As currently used, however, none of these remedies is completely suitable for flight control applications. Instead, approaches based on the use of *localized connectionist learning systems* have been developed as described below.

A.3.2 Learning-topology

The speed of learning and the quality of the mapping that is learned is a function of a variety of factors. The first is the learning algorithm (e.g., backpropagation as described above). Another factor that is nearly as important as the learning algorithm is the network topology, i.e., the number of layers in the network, the interconnections among nodes and the number of nodes in each layer. Too few layers/nodes results in an underdetermined problem. Too many layers/nodes can result in an overdetermined problem. A number of approaches to searching for good network topologies have been investigated and some are described in [A-2].

A.4 Applications to GN&C Work Systems

The purpose of the descriptions of the applications of neural networks presented here is that, for any given complex problem, neural networks play a role as ***an element of the solution*** to that problem and that it is rarely appropriate to simply attempt to have the network "blindly learn" to solve the entire solution to the problem by presenting the network with a sufficient number of suitable [problem state[1] : problem solution]. Furthermore, when attempting to apply learning systems to any problem, one must attend to the inherent practical limitations discussed above that are related to network learning rates, learning generalization and network topology selection.

A.4.1 Application of Neural Networks to Flight Control

Section 4.5 reviews some of the challenges encountered in the design of flight control systems, describes the some of the limitations of traditional approaches to flight control system design that are encountered when addressing these challenges and highlights the role and potential benefits of connectionist systems when applied to overcome those limitations. It is important to emphasize that no single, systematic approach for the application of connectionist learning systems to control has yet to evolve. The objective of this section is to introduce a few of the emerging approaches for learning control and to provide some references where further discussions may be found.

---

1 *Problem State* refers to the set of information that is required to completely characterize the problem at hand.

A.4.1.1 Connectionist Learning Systems for Control[2]

The ability of connnectionist systems to approximate smooth multivariable, nonlinear functions is the property that has generated a great deal of interest among control system theorists. *Copying or mimicking an existing controller* is perhaps the simplest approach to applying connectionist learning systems to control. Assuming there exists a controller that is able to control the plant, the objective of the connectionist learning system in this case would be to synthesize the mapping between the inputs and the desired output supplied by the existing controller (indeed, the existing controller may be in the form of a human operator). This approach may be useful in situations wherein it is beneficial for a learning system to replace an existing controller. This may pertain when the learning system offers a more efficient implementation or when the network can learn to mimic a control law that had heretofore been embodied in a human implementation and replace the human in the loop. This approach has been successfully applied to a pole balancing problem by Widrow and Smith (1964), where the existing control was supplied by a human.

Direct inverse control is another method which can benefit from the application of a connectionist learning system to control (Werbos (1989)). For this approach, the objective of the learning system is to identify and model the system inverse. This is accomplished by providing an observed output of the plant as the input to the network, and the observed input to the plant (i.e., the control signals) is provided as the desired network output. If a unique plant input produces a unique plant output, then when the desired plant output is provided as input to the network, the resulting network output is the control to be used as input to the plant (Barto (1989)). The drawbacks to this technique are that a desired reference trajectory and associated plant inputs must be known in order to supply the network with the desired plant input-output pairs and the inverse of the plant must be well-defined (e.g., a 1-to-1 mapping between inputs and outputs).

In the *backpropagation through time* method developed by Jordan (1988), two connectionist learning systems are used. The objective of the first network is to identify the plant, from which one can efficiently compute the derivative of the model output with respect to its input by means of back-propagation. Subsequently, propagating errors between the actual and a desired plant output back through this network produces an error in the control signal which can be used to train the second network (Barto (1989)). This approach offers an improvement over direct inverse control since it is able to accommodate systems with ill-defined inverses, although the desired trajectory must still be known.

Another approach for incorporating learning into a control system is to augment an adaptive technique with a learning system to form a *hybrid controller* (Baker & Farrell (1990), Baird (1991), Nistler (1992)). Augmentation of the adaptive technique can be implemented using either a direct or an indirect approach. Using a *direct* adaptive approach, the learning system generates a control action (or set of parameters) associated with a particular operating condition (thus, a representation of the operating condition is the network input and the control action is the output). This control action is then combined with an auxiliary control action produced by the adaptive system to arrive at the control that is applied to the plant. In contrast, for the *indirect* approach, the objective of the learning system is to improve the model of the plant. Here, the learning system generates model parameters that are a function of the operating condition. The parameters are combined with adaptive component estimates to arrive at a model of the plant. Given a presumably improved plant model, an on-line control law design is used to dynamically modify the closed-loop system. These two approaches are discussed in further detail below.

Reinforcement learning has also been suggested as a method of applying connectionist learning systems for control (Mendal & McLaren (1970), Barto (1989), Millington (1991), Barto (1992??)). The major difference between reinforcement learning and the previously discussed approaches is that under reinforcement learning, the objective is to optimize the overall behavior of the plant, so that no reference / desired trajectory is required. As a result, reinforcement learning essentially involves two problems, the construction of a *critic* that is capable of evaluating plant performance in a manner that is consistent with a stated control objective and the determination of how to alter the controller outputs to improve performance as measured by the critic (Barto (1989)). The latter of the problems can be addressed by one of the previously discussed techniques.

---

[2]This section is largely derived from material in [C-14,15,16]

1 Learning Augmented Direct Adaptive Control

Model reference control is a direct adaptive method (i.e., one that does not explicitly rely on on-line model identification) whose architecture is well-suited to the incorporation of learning (see for example [A-7]). In contrast to conventional model reference adaptive control techniques whose objective is to update the parameters of the control system fast enough to provide *local compensation* for varying dynamical behavior, the objective of the model reference *learning* control approach is to adjust the learning system parameters so as to develop a *globally applicable control law* that directly accommodates state dependencies and nonlinear dynamics. A small number of the network parameters can also be updated at higher learning rates, so that the system as a whole provides adaptation as well as learning, thereby accommodating both time-varying and nonlinear (state-dependent) dynamical behavior.

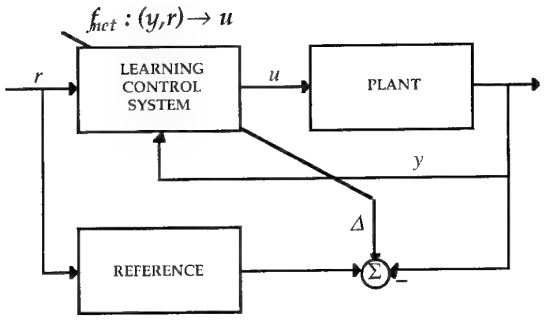


Figure A.4 Learning Augmented Model Reference Adaptive Control System Architecture. The neural net helps to learn the direct mapping from the reference value r to the appropriate control u .

For the model reference learning control architecture shown in Figure A.1-4, the performance of the closed-loop system is evaluated by comparing the actual response of the plant to the response of a reference system. In this case, the control system is a connectionist network that constructs the nonlinear control law required to force the closed-loop system to match the desired behavior of the reference system. That is, the network learns the nonlinear mapping $f_{net} : (y, r) \rightarrow u$, from the plant output y and reference input r to the control action u, that is needed to obtain the reference behavior. *As usual, care must be taken to ensure that the closed-loop system is physically capable of achieving the level of performance specified by the reference system.*

2 Learning Augmented Indirect Adaptive Control

Hybrid control is an indirect method that explicitly performs on-line model identification (see [A-5, A-6] for a more thorough description). In this scheme, an indirect adaptive control technique is coupled with a connectionist learning system to provide real-time adaptation to time-varying dynamics and disturbances, in conjunction with on-line learning to accommodate quasi-static state dependencies and nonlinearities. The adaptive control system reacts to discrepancies between the anticipated and observed behaviors of the plant in order to maintain the requisite closed-loop system performance. These discrepancies arise from unmodeled or time-varying dynamics, noise, and disturbances, and have a temporal or state-dependent basis. The learning system is used explicitly for the purpose of accommodating the state-dependent component of the discrepant behavior. Initially, all discrepancies are handled by the adaptive system; eventually, however, the learning system is able to learn and thereby *anticipate* previously experienced state dependencies and convey this information to the adaptive controller. Thus, the adaptive component of the system is able to focus on reacting to time-varying dynamics and disturbances and is not continuously burdened with the task of reacting to state- dependent nonlinear effects.

A schematic of one possible realization of an indirect hybrid adaptive learning control system is shown in Figure A.5. The indirect adaptive controller generates a control action u based upon the plant output y, the reference input r, and an estimate of the current dynamical behavior of the plant, which is derived from an internal model having a parameter vector μ. If the behavior of the plant changes (e.g., due to nonlinear or time-varying dynamics), the estimator (i.e., the model identification scheme) within the adaptive controller will attempt to update its model of the current plant dynamics by adjusting μ. Note that in the absence of learning augmentation, the adaptive controller would take as long to compensate for predictable state dependencies as it would for unpredictable effects such as time-varying external disturbances.

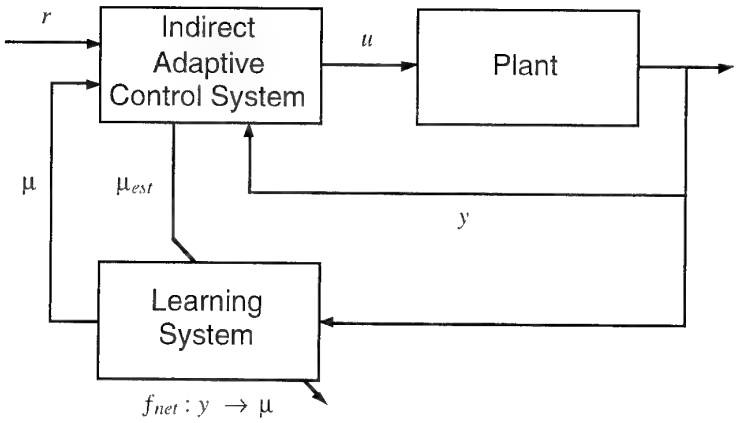


Figure A.5 Learning Augmented Indirect Adaptive Control System Architecture. In this approach the neural network helps to improve the plant model, providing updates of the model parameters μ.

A.4.1.2 Localized Learning

The concept of localized learning is one key to realizing real-time on-line learning for flight control. The underlying idea is based on the observation that learning can be simplified in situations where a direct association can be made between a subset of the adjustable elements of the learning system and a localized region of the input-space. Under such conditions, the effects of incremental learning on the overall mapping are limited to a localized region of the mapping. Thus, experience and consequent learning in one part of the input-space has only a marginal effect on the learning that has already occurred in other regions of the input-space. Locality is an intrinsic property of learning systems that employ quantized (bin) mappings; it is *not* a natural property of the standard Multi-Layer Perceptron neural networks in widespread use. Radial basis function networks [A-2] and Gaussian basis function networks [A-9] both exhibit this property of localized leaning. The latter is described below.

The approach developed in [A-10] and employed in [A-9] relies on a combination of *basis* function and *influence* function nodal units to achieve a compromise between the state-dependent localized learning properties of quantized learning systems and the efficient representation and generalization capabilities of standard artificial neural networks. The complete network mapping is constructed from a set of not necessarily orthogonal basis functions, that have applicability only over localized regions of the input space. The influence functions are coupled to the basis functions and are used to describe the domain (over of the input-space) of the basis functions. The key features of this approach are: (i) generalization is an inherent property of the network, and (ii) standard gradient learning methods can be utilized.

To illustrate the basic concept, consider a specific realization (many are possible) employing linear units (hyperplanes) as basis functions, and multivariable Gaussian units (hyper-Gaussians) as influence functions. The linear-Gaussian network is well-suited to the problem of synthesizing a control law. For example, this learning system could be used as an automatic, self-optimizing, gain scheduling control system. Because of its unique structure, physical meaning may be attributed to each parameter and to the overall structure of the network. As a result, *a priori* knowledge and partial solutions are easily incorporated (e.g., linear control "point" designs). In fact, hyperplanes are a good choice for the basis functions due to their simplicity and compatibility with conventional gain scheduled mappings (alternative basis functions may be more desirable if certain *a priori* knowledge is available about the regional functional structure of the desired solution). In addition, a linear-Gaussian network structure allows implementation of on-line variable structure learning, where nodal units can be added or removed from the network to achieve better or more efficient mappings. In contrast, there is no easy way to encode *a priori* knowledge or employ on-line variable structure learning with standard artificial neural networks.

A.4.2 Application of Neural Networks to Image/Pattern Recognition

Probably the most widespread use of neural networks is in pattern recognition and classification problem. A typical image recognition applications involves the design of preprocessing, feature extraction, and classifier functions. Image preprocessing is designed to enhance the desired signals and reduce the noise. Features are chosen

to minimize the ***intra***-class separation and maximize the ***inter***-class separation of their probability distribution function. The classifiers considered here consist of a multi-layer Perceptron (MLP) neural network and a thresholding function as depicted in the last two blocks of processing in Figure A.6.

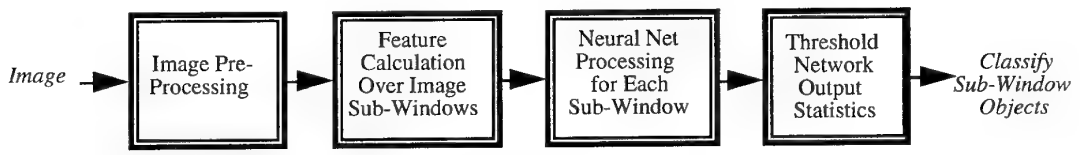


Figure A.6 Steps in Image Classification.
Rather than having the Neural Network operate directly on the image pixels, features are first extracted and used as inputs to the neural network. This considerably reduces the complexity of the neural network and the associated learning problem.

Two problems arise in designing a statistical pattern recognition classifier. One is the design of a classifier that takes input features and maps those features into classes and the other is the selection of the features that are used as the input. The ease of design and implementation of a neural network based classifier makes the problem of feature set selection much simpler. That is, for each candidate feature set a neural net can be trained to optimally process those features. The performance of the candidate features sets can then be ranked and the best candidate feature set chosen for the final system design.

A.4.2.1 Neural Network Approaches to Multi-sensor Fusion

Innovative approaches to multi-sensor fusion can be developed that utilize neural networks. Recently, the design and implementation of neural network architectures which are referred to as "hierarchical" networks have been pursued. The networks N1 and N2 in Figure A.7 are components of a simple hierarchical architecture. Network N1 in the figure functions as a screener; it is trained to make high confidence decisions and pass on other data to N2 for final resolution. Network N2 is tuned to a subset of the features extracted from images and needs fewer connections to learn the appropriate classification decisions given this small input set along with the output of Network N1. In this manner a complex set of posterior probability distributions can be learned without over parameterization. Thus, sequential networks can be linked in larger sets to decompose a difficult problem.

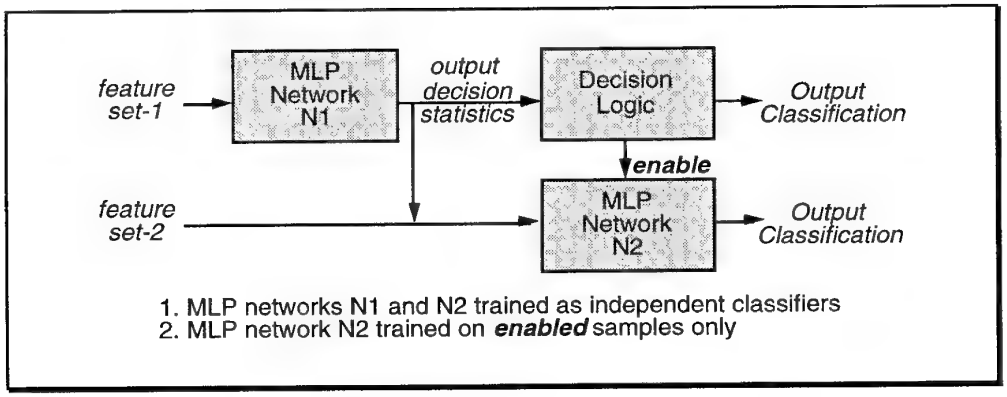


Figure A.7 Sequential Networks.
Sequential neural networks can be used to sequentially divide feature space in an optimal fashion. They can also be implemented to recognize temporal sequences.

The networks in Figure A.8 represent another simple hierarchical architecture. Each network in the figure is trained independently on different features sets that have been extracted from the same image data. Thus, each network can be viewed as a separate *knowledge source.* Knowledge is correlated or fused in the combining network N12. The architecture in effect functions as an information fuser. For example, distinct multiple sensor data can be fused in combining networks. The designer has the freedom to choose what is to be taught at each output node.

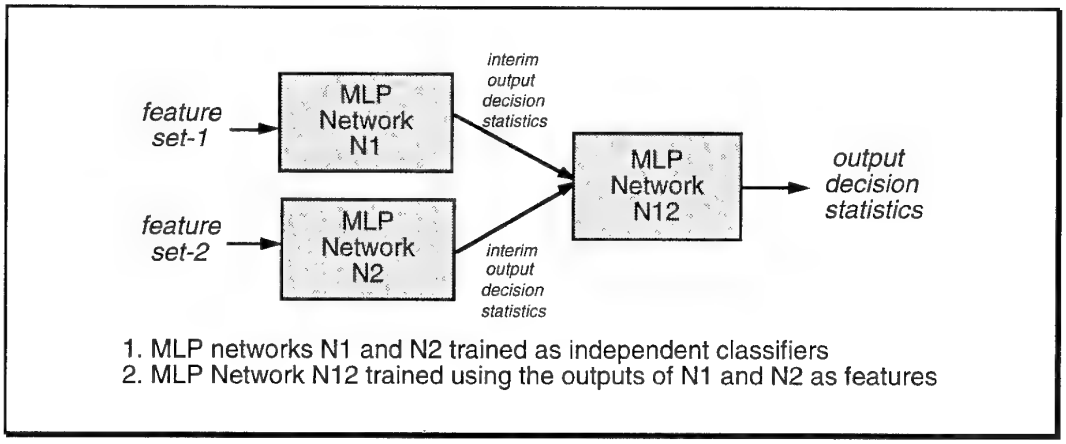


Figure A.8 Combining Networks.
In the hierarchical network depicted here, each network N1 and N2 is trained independently on the same data set but using different feature sets. The combining network N12 then fuses the outputs of N1 and N2 to make a more accurate decision.

Other simpler structures are suggested by combining networks. Sensor fusion at the decision level, for example, can be implemented with AND or OR logic. Other variants of this approach, using different logical operators are also possible. Combining-networks of the kind described here can be taught situation dependent mappings of multi-sensor data just as sequential networks can be trained to fuse information temporally.

A.4.3 Application of Neural Networks to Mission Planning

Some of the initial results that were obtained in applying recurrent (Hopfield) networks to traveling salesman planning problems [A-11] appeared promising. Those applications intended to capitalize on the recurrent network's ability to seek the minimum of the class of functions that could be mapped onto the energy function of the network (stable dynamical systems tend to seek a state of minimum energy). However, those approaches proved problematic when attempts were made to scale them up from simple traveling salesman problems to real world mission planning scenarios wherein uncertainty, constraints and exogenous factors such as threats dominate. The application described below is more in the spirit of the previously discussed applications of neural networks in that the neural network is applied in solving a part of the problem, taking advantage of the network's ability to approximate (learn) an unknown function. In this case, the function to be learned is that which selects appropriate search heuristics at each stage of an iterative search for the solution to the planning problem.

A.4.3.1 The Mission Planning Problem

The military aircraft mission and trajectory planning problem is that of planning the activities of an aircraft during its mission in a manner that best satisfies specified mission objectives while also satisfying mission and operational constraints. The activities that must be performed by an aircraft during the course of a mission can be broken down into the following categories [A-12].

(1) **Preflight Check**
(2) **Takeoff and Climbout**
(3) **Ingress**
(4) **On-station Activity**
(5) **Egress**
(6) **Descent and Landing**
(7) **Post-flight Debriefing**

Note that (3) and (4) may be performed several times in sequence for missions with more than a single mission objective.

Of the seven activities, the most critical, both in terms of mission success and in terms of the of planning requirements are ingress, and on-station and egress. These activities require two types of planning. The first type of planning concerns selecting the subset of mission objectives to be pursued and generating the aircraft's flight path.

The second type of planning concerns planning the activity of the aircraft's sensor, weapon and ECM systems along the flight path during both the ingress/egress and the on-station phases of the mission. Two neural-network-based approaches to augmenting the planning activity associated with selecting and ordering mission objectives are discussed. First, a brief definition of the mission and trajectory planning problem is given.

A ***mission plan*** consists of (a) an ordered sequence of objectives satisfying a specific set of constraints along with (b) a set of trajectories defining the flight path to be taken between each pair of objectives. Consequently, a natural method of approaching the aircraft mission and trajectory planning problem is to decompose the problem into two parts which will be referred to here as the ***Trajectory Planning Problem*** and the ***Mission Planning Problem***.

A.4.3.2 The Search for a Mission Plan: An Overview

The approach to iteratively generating on-board and in real time a mission plan that is described here is based on the approach first introduced in [A-13]. It starts with an initial plan consisting of only the departure state (location) and the recovery goal (location) and, if desired, other intermediate objectives as well. This initial plan is iteratively modified using a heuristic search in an attempt to generate, within a specified time available for plan generation[3], a plan of maximum expected utility (or value) [A-14]. The procedure is an iterative improvement scheme wherein for each iteration heuristics are employed to guide the search in an attempt to improve upon the current solution. Each iteration consists of the six steps illustrated in Figure A.9 and described individually below.

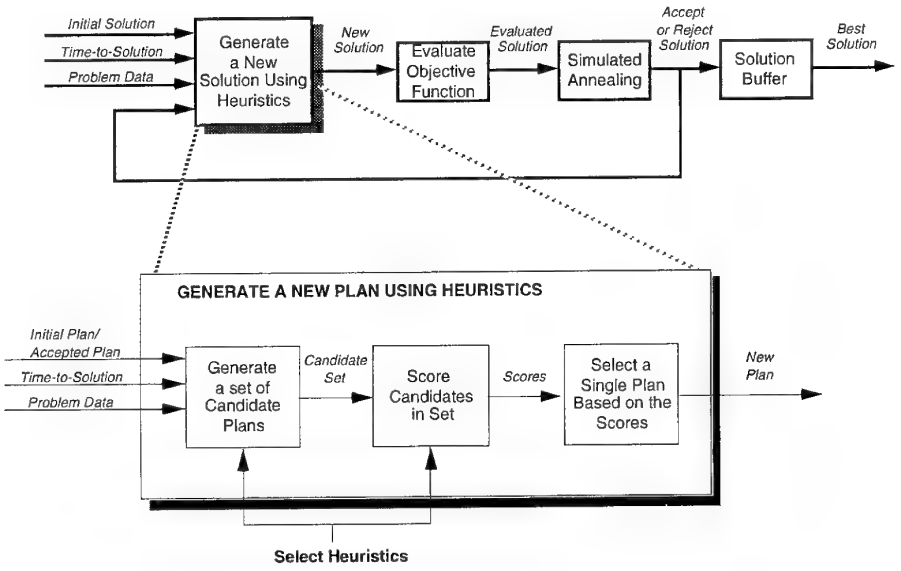


Figure A.9 The Heuristic Search Process.
The heuristic search is a six step process consisting of (1) heuristic selection, (2) trial set generation, (3) trial set scoring, (4) trial plan selection, (5) trial plan evaluation, and (6) trial plan acceptance or rejection.

The six steps of the mission planning cycle are repeated until the time allotted for planning has been exhausted or a point of diminishing returns has been reached. Of course, heuristic iterative improvement does not guarantee that the optimal plan will be found in the available planning time. However, in situations where there is sufficient time to plan, experience has shown that the planner is likely to find optimal or near-optimal solutions. More importantly, in situations where there is insufficient time to find the optimal solution, this approach produces acceptable, often high quality, solutions. This is an important and necessary attribute for real-time applications.

---

3 The coordination function as defined in Section 2.1 determines this specification

Select Plan Generation Heuristics and Generate a Set of Candidate Plans: The first step of each cycle of the process is the generation of a set of trial or candidate perturbations to the current plan in an attempt to produce at least one plan of greater value. Simple perturbations are created, for instance, by adding, removing, or reordering goals and by substituting new trajectories for those contained within the current mission plan. The nature and type of perturbations to the current plan during each cycle are selected heuristically. For instance, when available onboard fuel is nearly exhausted by the current plan, heuristics would tend not to introduce perturbations that would cause an increase in fuel consumption. Thus, rather than generating all possible perturbations (a potentially very large number), the heuristics tend to select perturbations that honor constraints while improving the value of the perturbed plan.

Select Plan Scoring Heuristics and Score Candidates in the Set: The members of the trial set are scored to provide a basis for selecting a single member of the trial set as the trial plan. Heuristics used in scoring the trial plans account for factors in the scoring process such as fuel use, probability of survival ($1 - P_k$), and effectiveness in achieving time windows. By controlling the scoring, the heuristics guide the search process. Trial plan scoring heuristics are selected based on the current planning situation and are implemented as a set of importance factors that indicate the relative importance of various resources (e.g., fuel, time, P_k) given the current plan. For example, if the current plan uses most of the available fuel, the weight on fuel would be set to a large value. Each trial plan of the trial set is scored.

Select a Single Plan Based on the Scores: The selection of a single trial plan is based on the scores assigned to each plan in the trial set. Since the perturbation selection and scoring heuristics do not guarantee that the best plan at any given cycle of the plan generation process will be created and receive the highest score, it would be unwise to overlook plans in the trial set with relatively high scores by selecting the plan with the highest score. Thus, members with the highest scores have the greatest probability of being selected, but plans with lower scores are not absolutely excluded from consideration.

Evaluate the Trial Plan: Once a new trial plan has been selected, it is more thoroughly evaluated on the basis a utility function. The utility is a single quantitative scalar measure for comparing the quality of any two mission plans.

Accept or Reject the Trial Plan via Simulated Annealing: The final step of the planning cycle is to decide if the perturbation to the original plan represented by the trial plan should be accepted or rejected. This can be done probabilistically via a simulated annealing like procedure to allow for the potential of escaping local minima in the plan search process.

A.4.3.3 The Use of Learning in Mission Planning

The goal of learning in the context of the approach to mission planning described above is to discover associations between planning strategies and planning situations that reduce the computational effort required in the search for good plans and thereby reduce the time required. Specifically, the objective is to learn for any given planning situation which *perturbation types* should be selected for use in generating the candidate set of trial plans and which *importance factors* should be selected for use in scoring the trial plans. Thus, these selection heuristics, which control the search strategy, are the object of the neural network learning. As described below, this results in learning situation- action pairs where the situation is the input to the network and the output is the action of selecting perturbation types and importance factors.

A *planning state space* is used to represent the spectrum of planning situations which are used as input. The dimensions of the state space correspond to features of both the planning environment (e.g., the time available for planning) and the plan that is to be modified (e.g., the fuel required and the risk assumed in executing that plan) that are pertinent to the selection decision. Since it is practical to consider only a small number of dimensions for the planning state space, a crucial design decision is the choice of which features are to be represented. Examples of

important feature are the amount of fuel that is consumed by the plan and the probability of survival for that plan. The state space is discretized by partitioning each of the feature dimensions into discrete intervals.

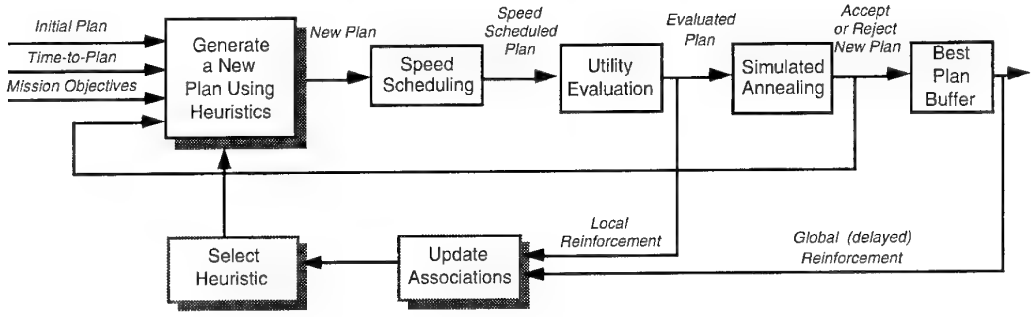


Figure A.10 Goal-point Planning Using Associative Reinforcement Learning.
Associative Reinforcement Learning is used to determine the parameters associated with the plan perturbation and scoring heuristics.

Figure A.10 shows feedback for learning that occurs during each iteration (*local reinforcement*) and delayed feedback (*global reinforcement*) occurring at the completion of the plan generation process. A feedforward neural network can be used to apply that feedback to learn the associations between good heuristics and the given planning situation. An alternative is the so-called *temporal difference learning* approach of Sutton [A-15] employed in [A-16] for learning strategies for playing the game of backgammon. Figure A.11 illustrates an approach which uses two networks: one to learn the appropriate plan perturbation selection heuristic (e.g., add a goal, remove a goal) and the other to learn the heuristics for selecting importance factors for scoring trial plans. Note that the selection of importance factors depends both on the planning situation and the type of perturbation to the plan that has been generated.

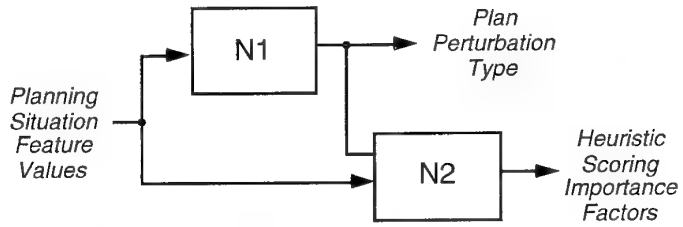


Figure A.11 Learning Plan Building Strategies.
The appropriate heuristic scoring factors will depend both on the planning situation and the select plan perturbation type.

A.5 References

[A-1] Rumelhart, D.,, G., and R. Williams, *Parallel Distributed Processing: Explorations in the of Cognition, Vol. 1: Foundations*, MIT Press / Bradford Books, 1986.

[A-2] Hush, D. R. and Horne, B. G., "Progress in Supervised Neural Networks: What's New Since Lippmann?" *IEEE Signal Magazine, Vol. 10, No. 1, pp. 8-39, January 1993.*

[A-3] Hornik, K., Stinchcombe, M., and H. White, "Multi-layer Feedforward Networks are Universal Approximators," *Neural Networks*, Vol. 2, 1989.

[A-4] Rumelhart, D., Hinton, G., and R. Williams, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*, MIT Press / Bradford Books, 1986.

[A-5] Baird, L. and W. Baker, "A Connectionist Learning System for Nonlinear Control," *Proceedings, AIAA Conference on Guidance, Navigation, and Control*, August, 1990.

[A-6] Baker, W. and J. Farrell, "Connectionist Learning Systems for Control," *Proceedings, SPIE OE/Boston '90* (invited paper), November, 1990.

[A-7] Farrell J., Goldenthal B., and K. Govindarajan, "Application of Neural Networks to Automatic Control: Submarine Heading Control," *Proceedings, 29th IEEE Conference on Decision and Control*, December, 1990.

[A-8] Goldenthal, B. and J. Farrell, "Application of Neural Networks to Automatic Control," *Proceedings, AIAA Conference on Guidance, Navigation, and Control*, August, 1990.

[A-9] Nistler, N. (1992). "A Learning Enhanced Flight Control System for High Performance Aircraft," CSDL Report T-1127, M.S. Thesis, Department of Aeronautics and Astronautics, M.I.T.

[A-10] Baker, W. & Farrell, J. (1992). "An Introduction to Connectionist Learning Control Systems," in Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches, White, D. & Sofge, D., eds., V. N. Reinhold.

[A-11] J.J. Hopfield and D. W. Tank, "Neural Computation of Decisions in Optimization Problems," *Biological Cybernetics*, Vol. 52, No. 3, pp. 141 - 152, 1985.

[A-12] Hunt, G. and Jones, T. editors, "Mission Planning Systems for Tactical Aircraft: Interim Report," AGARD, 1991.

[A-13] M. B. Adams, O. L. Deutsch, and J. V. Harrison, "A Hierarchical Planner for Intelligent Systems," *Proceedings of SPIE - The International Society for Optical Engineering*, Vol. 548, Arlington, VA, April 9-11, 1985.

[A-14] Beaton, R. M. , M. B. Adams, and J. V. Harrison, "Real-Time Mission and Trajectory Planning," *Proceedings of the 26th IEEE Conference on Decision and Control*, Los Angeles, CA, December, 1987.

[A-15] Sutton, R.S., "Learning to Predict by the Method of Temporal Differences," *Machine Learning*, Vol. 3, pp. 9-44, 1988.

[A-16] Tesauro, R., "Practical Issues in Temporal Difference Learning," *Machine Learning*, Vol. 8, pp. 257-277, 1992.

APPENDIX B

MATERIAL PRESENTED TO WORKING GROUP (WG) 11 DURING WG MEETINGS OR SITE VISITS

B.1 Working Papers prepared by Members of WG 11

The following Working Papers have been written by Members of WG 11 and have been discussed at the 6 Meetings of the Working Group:

H. Winter: The Impact of Traditional and New Software Technologies on the Life Cycle of G&C Systems

H. Kanoui: Applicable Technologies

M. Adams: Functional Decomposition of Onboard Intelligent Systems

M. Adams: Application of Neural Networks for GN&C

P. Friedland: Artificial Intelligence Technology Development Plan

P. Friedland: Operational Efficiency Subpanel: Advanced Mission Control

P. Friedland: AI Technology Overview (Collection of VG)

G. Champigneux: Functional Analysis and Architecture of G&C of Aircraft

D. Holmes: Intelligent System Functionality in Modern Aircraft

E. Girardelli: Functional Analysis and Architecture of Pre-Mission Activities

F. Kraiss: Human in the Loop

K. Rosenberg: Glossary of Terms for Knowledge Based Systems

R. Onken: Baseline Functional Structure of Pilot Assistant Systems

D. Holmes: Functional Analysis of G&C in Modern Military AC

D. Godart: Real-Time On-board Expert System for Navigation and Guidance

P. Friedland: Enabling Technologies and their Potential

H. A. Funk: Effects of Applications of AI on Life Cycle Costs

K. Rosenberg: Information Technologies Review

G. Champigneux: Task Allocation between a Pilot and a Crew Assistant System

R. Hood: Enabling Technologies and their Potential

B. Ellis: Functional Allocation to Man and Machine

S. Sallé: Real-time On-board Expert System for Navigation and Guidance (Copies of VGs)

S. Sallé, G. Champigneux and D. Holmes: Developing Environment for Knowledge Based Systems used in G&C.

H. A. Funk: Live Cycle (Collection of VGs)

P. Friedland and H. A. Funk: Enabling Technologies and their Potential .

M. Adams: Control System Design Methodologies.

H. Kanoui: Enabling Technologies

M. Adams, G. Champigneux and D. Holmes: Top-Level Functional Structure (Collection of VGs)

J. M. Darroy: Knowledge-based Spacecraft Operations (Collection of VGs)

U. Völckers: Functional Analysis and Architecture of G&C in ATC (Collection of VGs)

B.2 Motivating Examples of Knowledge Based System Applications Discussed by WG 11

Leading Experts in the field of knowledge based G&C systems have been invited to discuss their work with WG 11 during the 6 meetings of the Working Group. The following documentation of this work was handed out to WG 11:

R. Onken, Universität der Bundeswehr, Mûnchen: Knowledge-Based Cockpit Assistant for IFR Operation (Collection of VGs)

D. Morillon, T. Conter et M. de Cremiers, SAGEM, Osny: SEAN: An Expert System for Navigation Assistance aboard Fighter Aircraft (Working Paper presented by S. Sallé)

N. Geddes, Search Technology, Inc., Norcross: The Lockheed Pilot's Associate Program (Collection of VGs)

W.B. Rouse, N.D. Geddes and R.E. Curry: An Architecture for intelligent interfaces: Outline of an Approach to Supporting Operators of Complex Systems. Human-Computer Interaction, 1987-1988, Volume 3 pp. 87-122.

W.B. Rouse, N.D. Geddes and J.M. Hammer: Computer-aided fighter pilots. IEEE Spectrum, March 1990.

D.R. Sewell, N.D. Geddes and W.B. Rouse: Initial Evaluation of an Intelligent Interface for Operators of Complex Systems. Proc. Second International Conference on Human-Computer Interaction, Honolulu, August 1987.

B.H. Hoshstrasser and N.D. Geddes, Search Technology, Inc., Norcross: OPAL - Operator Intent Inferencing for Intelligent Operator Support Systems.

B.W. Webb, N.D. Geddes and L.O. Neste: Information Management with a Hierarchical Display Generator. Proc. National Computer Graphics Assoc. April 1989

N.G. Geddes, J.M. Hammer: Automatic Display Management Using Dynamic Plans and Events. 6th Symposium on Aviation Psychology, Columbus, April 1991.

V.L. Shalin, et al.: Towards a Theory of Pilot Information Requirements During Plan Development and Execution. Proc. of CERT 90, France, Sept. 1990.

J.D Hammer and N.D. Geddes: Design of an Intelligent Monitor for Human Error in a Complex System. Computers in Aerospace 6, Wakefield, October, 1987. AIAA.

R.C. Andes: Adaptive Aiding in Complex Systems: An Implementation. 1987 Proc. of IEEE Systems, Man, and Cybernetics Society Conference.

D.R. Sewell and N.D. Geddes: A Plan & Goal-Based Method for Computer-Human System Design. Human-Computer Interaction INTERACT-90 North Holland. Diaper, Gilmore, Cockton and Shackel (Eds.) 1990.

D.J. Atkinson, Jet Propulsion Laboratory, Pasadena: SHARP: Spacecraft Health Automated Reasoning Prototype (Collection of VGs)

A. Benoit, EUROCONTROL, Brussels: Knowledge Based Systems in ATC (No Paper Handed Out)

E.L. Duke, R.W. Brumbaugh, M.D. Hewett and D.M. Tartt, NASA Dryden Flight Research Facility, Edwards, Cal.: From an Automated Flight-Test Management System to a Flight-Test Engineers Workstation.

J.W. McManus, NASA Langley Research Center: Knowledge-Based G&C (Collection of VGs)

B. Sridhar, NASA Ames Research Center: Application of Computer Vision Techniques to G&C. (No Paper Handed Out).

S. Damiani, F. Ricci and E. Valfré, Aliena Aeronautica, Torino, Italy: ADAM Aircraft Diagnostic And Maintenance (Working Paper presented by E. Girardelli).

B.3 Applications of Knowledge Based Systems Discussed During Site Visits of WG 11

In connection with the Meetings of WG 11 several Site Visits have been made in order to study successful examples of knowledge based systems in G&C. The Documentation of the work of the visited organizations handed out to WG 11 is listed in this appendix. The Site Visits took place in Germany on the 13th and 14th May, 1991, in France on 28 and 29 October 1991, and in USA on 19, 21 and 22 May, 1992.

B.3.1 Visits in Germany

The WG visited **Frankfurt Airport** to see the COMPAS system developed by DLR (see Section 6.1.4) which has been in operation at this airport since 1989. In Munich the Group visited the **Siemens AG** at Unterschleissheim where presentations were made describing activities in air traffic management, an expert system for real time

applications, and neural networks for data fusion. At the **Universität der Bundeswehr,** the CASSY system (see Section 6.1.1) and the work on autonomous vehicle guidance by computer vision were presented to the Group. The **Forschungsinstitut fûr Anthropotechnik** (FGAN-FAT), Werthoven, Germany had organized a presentation in Munich describing neural operator models in a car driving situation. At **MBB** the Threat Management System, the Airfield Management System, and facilities for man/machine interface prototyping were presented to the Group. The following documentation was obtained at these visits.

DLR COMPAS System at Frankfurt Airport

Computer Assisted Arrival Planning (Collection of VGs)

COMPAS: A Planning and Decision Aid for Air Traffic Controllers (DLR Brochure)

Siemens, Unterschleissheim

Application of Neural Networks (Collection of VGs)

Recognized Air Picture Production with Expert Systems (Collection of VGs)

Echtzeit-Expertensysteme (Folder in German)

Realzeit-Expertensystem: Battleman (Collection of VGs and Folder in German)

Universität d. Bundeswehr, Munich

E.D. Dickmanns and V. Graefe: Dynamic Monocular Machine Vision. UniBwM/lrt/WE 13/FB/88-3.

Th. Wittig: Knowledge-Based Cockpit Assistant for IFR-Operations (Collection of VGs)

R. Onken: Knowledge-based Cockpit Assistant for IFR-Operations. AGARD GCP 51st Symposium, Madrid, Sept. 1990.

M. Kopf and R. Onken, Universität der Bundeswehr, Mûnchen: A Machine Co-Driver for Assisting Drivers on German Autobahns.

K.-F. Kraiss and H. KÅttelwesch, Research Institute for Human Engineering (FGAN-FAT), Werthoven, Germany: Identification and Application of Neural Operator Models in a Car Driving Situation.

K.-F. Kraiss and H. Küttelwesch: Teaching neural networks to guide a vehicle through an obstacle course by emulating a human teacher. Proc. of International Joint Conference on Neural Networks, IEEE S. Diego, June 1990.

E.D. Dickmanns and Th. Christians: Relative 3D-State Estimation for Autonomous Visual Guidance of Road Vehicles. Intelligent Autonomous Systems 2 (IAS-2), Amsterdam, Dec. 1989.

E.D. Dickmanns: Dynamic Vision for Intelligent Motion Control. IEEE-Int. Workshop on "Intelligent Motion Control", Istanbul, Aug. 1990.

MBB, Munich

K. Holla and B. Benninghofen: A Threat Management System.

Information System Supporting Survivability (Folder)

TMS - Threat Management System (Collection of VGs)

System Engineering and System Development Technologies at MBB (Collection of VGs)

List of AI - Projects at MBB

B.3.2 Visits in France

Visits were made to **Dassault Aviation** (Saint Cloud) and **CERMA** (Aerospace Medical Research Center) in Paris and to **MATRA Space** and **Dassault** in Toulouse. Discussions and demonstrations at Dassault were related to the "Electronic Co-Pilot" Program. Discussions at CERMA dealt with cognitive modeling and man/machine coordination. The visit to MATRA involved a number of presentations and discussions with members of the Artificial Intelligence Department. The topics covered included several applications of AI/Expert Systems to space missions. At Dassault in Toulouse, discussions centered around AI applications to industrial planning, scheduling, and computer-aided design. The following materials were received during these visits.

DASSAULT, Saint Cloud and Toulouse

Development Environment for the Copilote Electronique. Fuzzy Guidance. Mission Preparation for the Copilote Electronique. Copilote Electronique Architecture Design.

(4 Short Papers by Dassault Aviation)

Intent Recognition in the Copilote Electronique. (Collection of VGs)

Guidance: Fuzzy Logic. (Collection of VGs)

CERMA (Aerospace Medical Research Center), Paris

Amalberti et al. CERMA, France and JRC, Italy: Modeling Preferences in Process Control: The Importance of Metaknowledge.

R. Amalberti et F. Deblon, CERMA: Cognitive Modeling of Fighter AC Process Control: A Step towards an intelligent onboard assistance system.

MATRA Space, Toulouse

J.M. Darroy: Artificial Intelligence at Matra Marconi Space. (Collection of VGs)

J.M. Darroy, MATRA: Presentation of the AI Department of MATRA.

J.M. Darroy: AI at MATRA Espace: Industrial Applications of Expert Systems. 3rd International Congress of Informatics Associations, Rio de Janeiro, August 1990.

J.M. Darroy: Integrated Human-Machine Intelligence in Aerospace. Proc. of International Conference Human-Machine Interaction and Artificial Intelligence in Aeronautics and Space, Toulouse, September 1990.

J.M. Darroy: Knowledge-Based Systems for Spacecraft Control. First International Symposium on Ground Data Systems for Spacecraft Control, Darmstadt, Germany, June 1990.

J.M. Darroy: AI for Space: A strategic approach for more than just a new technology. 3rd European Space Agency Workshop, Noordwijk, May 1991 on AI and Knowledge-Based Systems for Space.

M. Nielsen et al.: Expert Operator's Associate: An expert system for spacecraft control. Proc. First International Symposium on Ground Data Systems for Spacecraft Control, Darmstadt, Germany, June 1990.

P. Caloud et al.: On-board Decision Support System for Orbital Rendez-Vous Operations. IMACS International Workshop on Qualitative Reasoning and Decision Support Systems, March 1991 in Toulouse.

F. Lecouat, M. Nielsen and K. Grue: Knowledge-Based assistance for Generating and Executing Space Operations Procedures. ESTEC Workshop on AI and Knowledge-Based Systems for Space. ESTEC, May 1991.

M. Haziza: Towards an operational fault operation expert system for French telecommunication satellite TELECOM 2. Proc. First International Symposium on Ground Data Systems for Spacecraft Control, Darmstadt, Germany, June 1990.

J.M. Brenot et al.: On the development of an operational expert system for the TELECOM 2 satellite control centre. ESTEC Workshop on AI and Knowledge-Based Systems for Space. ESTEC, May 1991.

J.J. Fuchs et al.: planERS-1: An expert planning system for generating spacecraft mission plans. First International Conference on Expert Planning Systems in Brighton, UK, June 1990.

J.J. Fuchs et al.: The Expert Planning System Prototype: Experimentation and Users Feedback. ESTEC Workshop on AI and Knowledge-Based Systems for Space. ESTEC, May 1991.

M.M. Arentoft et al.: OPTIMUM-AIV: A Planning and Scheduling System for Spacecraft AIV. ESTEC Workshop on AI and Knowledge-Based Systems for Space. ESTEC, May 1991.

J.-M. Brenot et al.: ARIANEXPERT: A Knowledge Based System to analyze ARIANE's mission data. ESTEC Workshop on AI and Knowledge-Based Systems for Space. ESTEC, May 1991.

R. Laurette et al.: Supervision and Control of the AMR Intervention Robot. Proc. of 5th International Conference on Advanced Robotics (ICAR), Pisa, Italy, June 1990.

M. Devy et al.: Terrain modeling from 3D depth images for an autonomous mobile robot. Proc. of 7th International Scandinavian Conference on Image Analysis, Aalborg, Denmark, August 1991.

B.3.3 Visits in USA

Visits were made to **NASA Johnson Space Center** in Houston, Texas and to **NASA Ames Research Center** in Mountain View, California. The demonstrations and discussions at the Johnson Space Center were related to the application of knowledge based system technology to spacecraft, payload, and space mission management. At the Ames Research Center visits were made to the AI, G&C, and ATC Laboratories. The demonstrations and presentations dealt with various AI technologies under development, with the application of knowledge based systems to air vehicle G&C, especially helicopter applications, and with automation of air traffic management functions, using computer intelligence.

NASA Johnson Space Center-Houston

G.J. Reuter et al.: An intelligent free-flying robot. SPIE Symposium on Advances in Intelligent Robotic Systems, Space Station Automation IV, Cambridge, MA, November 1988.

J. Ericson et al.: Technology test results from an intelligent, free-flying robot for crew and equipment retrieval in space. Proc. of Society of Photo-Optical Instrumentation Engineers Symposium on Cooperative Intelligent Robotic Systems in Space II, Boston, MY, November 1991.

J. Ericson et al.: Future needs for space robots for SEI. Proc. of Society of Photo-Optical Instrumentation Engineers Symposium on Cooperative Intelligent Robotic Systems in Space II, Boston, MY, November 1991.

J.L. Fox, NASA, LBJ Space Center, Houston: Space shuttle vehicle familiarization.

NASA, LBJ Space Center, Houston: Payload deployment and retrieval system.

NASA, LBJ Space Center, Houston: DESSY Screen Description (Collection of VGs)

NASA Ames Research Center,-Mountain View

G.A. Boy, NASA Ames Research Center: Indexing Hypertext Documents in Context.

H.R. Berenji: Treatment of uncertainty in AI. In Machine Intelligence and Autonomy for Aerospace Systems. Heer and Lum. Progress in Astronautics and Aeronautics. Vol. 115.

D. Clarc, et al.: Responses to "An AI view of the treatment of uncertainty" by Alessandro Saffotti. The knowledge engineering review. Vol. 3 No. 1. March 1988.

H.R. Berenji et al.: A hierarchical approach to designing approximate reasoning-based controllers for dynamic physical systems. Uncertainty in AI 6, 1991 Elsevier Science Publishers,

H.A. Berenji: A reinforcement learning-based architecture for fuzzy logic control. Intern. Journal of Approximate Reasoning. Vol 6, Nr. 2, February 1992.

M. Zweben et al.: Rescheduling with Iterative Repair. NASA Ames Research Center Technical Report FIA-92-15. April 1992.

H. Berenji et al.: Learning and tuning fuzzy logic controllers through reinforcements. NASA Ames Research Center Technical Report FIA-92-02. January 1992.

P.N. Smith et al.: Vision-Based Range Estimation Using Helicopter Flight Data. IEEE Conference on Computer Vision and Pattern Recognition, Champaign II, June 1992.

V. Cheng and B. Sridhar: Technologies for Automating Rotorcraft Nap-of-the-Earth Flight. AHS 48th Annual Forum, Washington, DC, June 1992.

V. Cheng and T. Lam: Automatic Guidance and Control Laws for Helicopter Obstacle Avoidance. IEEE Conf on Robotics and Automation, Nice France, May 1992.

H.N. Swenson et al.: Simulation Evaluation of a Low-Altitude Helicopter Flight Guidance System Adapted for a Helmet-Mounted Display. NASA Technical Memorandum 103883.

V. Cheng and B. Sridhar: Considerations for Automated Nap-of-the Earth Rotorcraft Flight. 1988 American Control Conference. Atlanta, June 1988.

B. Sridhar and Suorsa: Comparison of Motion and Stereo Methods in Passive Ranging Systems. IEEE Trans. on Aerospace and Electronic Systems. Vol 27, No. 4, July 1991.

H.N. Swenson: Computer Aiding for Low-Altitude Helicopter Flight. NASA Technical Memorandum 103861.

B. Sridhar et al.: Kalman Filter Based Range Estimation for Autonomous Navigation Using Imaging Sensors. Automatic Control in Aerospace. IFAC Symposium Tsukuba, Japan, July 1989.

H. Erzberger CTAS: Computer Intelligence for Air Traffic Control in the Terminal Area. Proc. Special Conference on "Air Traffic Control: Data Processing and Large Scale Computation". Italy's National Research Council (CNR), Capri, Italy, October 1991.

REPORT DOCUMENTATION PAGE

1. Recipient's Reference	2. Originator's Reference	3. Further Reference	4. Security Classification of Document
	AGARD-AR-325	ISBN 92-836-1009-1	UNCLASSIFIED/ UNLIMITED

5. Originator Advisory Group for Aerospace Research and Development
North Atlantic Treaty Organization
7 rue Ancelle, 92200 Neuilly-sur-Seine, France

6. Title
Knowledge-Based Guidance and Control Functions

7. Prepared at
The request of the Guidance and Control Panel of AGARD

8. Author(s)/Editor(s)	9. Date
Multiple	January 1995
10. Author's/Editor's Address	**11. Pages**
Multiple	192

12. Distribution Statement There are no restrictions on the distribution of this document. Information about the availability of this and other AGARD unclassified publications is given on the back cover.

13. Keywords/Descriptors

Automatic control	Airborne operations
Control equipment	Artificial intelligence
Guidance	Automation
Expert computer systems	Knowledge bases
Aircraft	Knowledge based systems

14. Abstract

This report summarises the deliberations of Working Group 11 of the Guidance and Control Panel of AGARD. The objectives of the Working Group were:

(1) Analyze the structure of knowledge-based guidance and control functions related to aircraft, missions, and the battlefield and identify their potential for automation.

(2) Analyze the structure of knowledge-based guidance and control functions related to the life cycle of guidance and control systems, and identify their potential for automation.

(3) Review the state-of-the-art of those software and hardware oriented technologies required for the transfer of the knowledge-based G&C functions to automatic systems.

(4) Review existing programmes.

(5) Make recommendations for future work.

AGARD Advisory Report No. 325 Advisory Group for Aerospace Research and Development, NATO KNOWLEDGE-BASED GUIDANCE AND CONTROL FUNCTIONS Published January 1995 192 pages This report summarises the deliberations of Working Group 11 of the Guidance and Control Panel of AGARD. The objectives of the Working Group were: (1) Analyze the structure of knowledge-based guidance and control functions related to aircraft, missions, and the battlefield and identify their potential for automation.	AGARD-AR-325 Automatic control Control equipment Guidance Expert computer systems Aircraft Airborne operations Artificial intelligence Automation Knowledge bases Knowledge based systems	AGARD Advisory Report No. 325 Advisory Group for Aerospace Research and Development, NATO KNOWLEDGE-BASED GUIDANCE AND CONTROL FUNCTIONS Published January 1995 192 pages This report summarises the deliberations of Working Group 11 of the Guidance and Control Panel of AGARD. The objectives of the Working Group were: (1) Analyze the structure of knowledge-based guidance and control functions related to aircraft, missions, and the battlefield and identify their potential for automation.	AGARD-AR-325 Automatic control Control equipment Guidance Expert computer systems Aircraft Airborne operations Artificial intelligence Automation Knowledge bases Knowledge based systems
AGARD Advisory Report No. 325 Advisory Group for Aerospace Research and Development, NATO KNOWLEDGE-BASED GUIDANCE AND CONTROL FUNCTIONS Published January 1995 192 pages This report summarises the deliberations of Working Group 11 of the Guidance and Control Panel of AGARD. The objectives of the Working Group were: (1) Analyze the structure of knowledge-based guidance and control functions related to aircraft, missions, and the battlefield and identify their potential for automation.	AGARD-AR-325 Automatic control Control equipment Guidance Expert computer systems Aircraft Airborne operations Artificial intelligence Automation Knowledge bases Knowledge based systems	AGARD Advisory Report No. 325 Advisory Group for Aerospace Research and Development, NATO KNOWLEDGE-BASED GUIDANCE AND CONTROL FUNCTIONS Published January 1995 192 pages This report summarises the deliberations of Working Group 11 of the Guidance and Control Panel of AGARD. The objectives of the Working Group were: (1) Analyze the structure of knowledge-based guidance and control functions related to aircraft, missions, and the battlefield and identify their potential for automation.	AGARD-AR-325 Automatic control Control equipment Guidance Expert computer systems Aircraft Airborne operations Artificial intelligence Automation Knowledge bases Knowledge based systems

(2) Analyze the structure of knowledge-based guidance and control functions related to the life cycle of guidance and control systems, and identify their potential for automation.

(3) Review the state-of-the-art of those software and hardware oriented technologies required for the transfer of the knowledge-based G&C functions to automatic systems.

(4) Review existing programmes.

(5) Make recommendations for future work.

ISBN 92-836-1009-1

(2) Analyze the structure of knowledge-based guidance and control functions related to the life cycle of guidance and control systems, and identify their potential for automation.

(3) Review the state-of-the-art of those software and hardware oriented technologies required for the transfer of the knowledge-based G&C functions to automatic systems.

(4) Review existing programmes.

(5) Make recommendations for future work.

ISBN 92-836-1009-1

(2) Analyze the structure of knowledge-based guidance and control functions related to the life cycle of guidance and control systems, and identify their potential for automation.

(3) Review the state-of-the-art of those software and hardware oriented technologies required for the transfer of the knowledge-based G&C functions to automatic systems.

(4) Review existing programmes.

(5) Make recommendations for future work.

ISBN 92-836-1009-1

(2) Analyze the structure of knowledge-based guidance and control functions related to the life cycle of guidance and control systems, and identify their potential for automation.

(3) Review the state-of-the-art of those software and hardware oriented technologies required for the transfer of the knowledge-based G&C functions to automatic systems.

(4) Review existing programmes.

(5) Make recommendations for future work.

ISBN 92-836-1009-1

AGARD

NATO OTAN

7 RUE ANCELLE • 92200 NEUILLY-SUR-SEINE

FRANCE

Télécopie (1)47.38.57.99 • Télex 610 176

DIFFUSION DES PUBLICATIONS

AGARD NON CLASSIFIEES

Aucun stock de publications n'a existé à AGARD. A partir de 1993, AGARD détiendra un stock limité des publications associées aux cycles de conférences et cours spéciaux ainsi que les AGARDographies et les rapports des groupes de travail, organisés et publiés à partir de 1993 inclus. Les demandes de renseignements doivent être adressées à AGARD par lettre ou par fax à l'adresse indiquée ci-dessus. *Veuillez ne pas téléphoner.* La diffusion initiale de toutes les publications de l'AGARD est effectuée auprès des pays membres de l'OTAN par l'intermédiaire des centres de distribution nationaux indiqués ci-dessous. Des exemplaires supplémentaires peuvent parfois être obtenus auprès de ces centres (à l'exception des Etats-Unis). Si vous souhaitez recevoir toutes les publications de l'AGARD, ou simplement celles qui concernent certains Panels, vous pouvez demander à être inclu sur la liste d'envoi de l'un de ces centres. Les publications de l'AGARD sont en vente auprès des agences indiquées ci-dessous, sous forme de photocopie ou de microfiche.

CENTRES DE DIFFUSION NATIONAUX

ALLEMAGNE
Fachinformationszentrum,
Karlsruhe
D-76344 Eggenstein-Leopoldshafen 2

BELGIQUE
Coordonnateur AGARD-VSL
Etat-major de la Force aérienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Bruxelles

CANADA
Directeur, Services d'information scientifique
Ministère de la Défense nationale
Ottawa, Ontario K1A 0K2

DANEMARK
Danish Defence Research Establishment
Ryvangs Allé 1
P.O. Box 2715
DK-2100 Copenhagen Ø

ESPAGNE
INTA (AGARD Publications)
Pintor Rosales 34
28008 Madrid

ETATS-UNIS
NASA Headquarters
Code JOB-1
Washington, D.C. 20546

FRANCE
O.N.E.R.A. (Direction)
29, Avenue de la Division Leclerc
92322 Châtillon Cedex

GRECE
Hellenic Air Force
Air War College
Scientific and Technical Library
Dekelia Air Force Base
Dekelia, Athens TGA 1010

ISLANDE
Director of Aviation
c/o Flugrad
Reykjavik

ITALIE
Aeronautica Militare
Ufficio del Delegato Nazionale all'AGARD
Aeroporto Pratica di Mare
00040 Pomezia (Roma)

LUXEMBOURG
Voir Belgique

NORVEGE
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25
N-2007 Kjeller

PAYS-BAS
Netherlands Delegation to AGARD
National Aerospace Laboratory NLR
P.O. Box 90502
1006 BM Amsterdam

PORTUGAL
Força Aérea Portuguesa
Centro de Documentação e Informação
Alfragide
2700 Amadora

ROYAUME-UNI
Defence Research Information Centre
Kentigern House
65 Brown Street
Glasgow G2 8EX

TURQUIE
Millî Savunma Başkanliği (MSB)
ARGE Dairesi Başkanliği (MSB)
06650 Bakanliklar-Ankara

Le centre de distribution national des Etats-Unis ne détient PAS de stocks des publications de l'AGARD.

D'éventuelles demandes de photocopies doivent être formulées directement auprès du NASA Center for AeroSpace Information (CASI) à l'adresse ci-dessous. Toute notification de changement d'adresse doit être fait également auprès de CASI.

AGENCES DE VENTE

NASA Center for
AeroSpace Information (CASI)
800 Elkridge Landing Road
Linthicum Heights, MD 21090-2934
Etats-Unis

ESA/Information Retrieval Service
European Space Agency
10, rue Mario Nikis
75015 Paris
France

The British Library
Document Supply Division
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
Royaume-Uni

Les demandes de microfiches ou de photocopies de documents AGARD (y compris les demandes faites auprès du CASI) doivent comporter la dénomination AGARD, ainsi que le numéro de série d'AGARD (par exemple AGARD-AG-315). Des informations analogues, telles que le titre et la date de publication sont souhaitables. Veuiller noter qu'il y a lieu de spécifier AGARD-R-nnn et AGARD-AR-nnn lors de la commande des rapports AGARD et des rapports consultatifs AGARD respectivement. Des références bibliographiques complètes ainsi que des résumés des publications AGARD figurent dans les journaux suivants:

Scientific and Technical Aerospace Reports (STAR)
publié par la NASA Scientific and Technical
Information Division
NASA Headquarters (JTT)
Washington D.C. 20546
Etats-Unis

Government Reports Announcements and Index (GRA&I)
publié par le National Technical Information Service
Springfield
Virginia 22161
Etats-Unis
(accessible également en mode interactif dans la base de données bibliographiques en ligne du NTIS, et sur CD-ROM)

Imprimé par le Groupe Communication Canada
45, boul. Sacré-Cœur, Hull (Québec), Canada K1A 0S7

AGARD

NATO OTAN

7 RUE ANCELLE • 92200 NEUILLY-SUR-SEINE

FRANCE

Telefax (1)47.38.57.99 • Telex 610 176

DISTRIBUTION OF UNCLASSIFIED

AGARD PUBLICATIONS

AGARD holds limited quantities of the publications that accompanied Lecture Series and Special Courses held in 1993 or later, and of AGARDographs and Working Group reports published from 1993 onward. For details, write or send a telefax to the address given above. *Please do not telephone.*

AGARD does not hold stocks of publications that accompanied earlier Lecture Series or Courses or of any other publications. Initial distribution of all AGARD publications is made to NATO nations through the National Distribution Centres listed below. Further copies are sometimes available from these centres (except in the United States). If you have a need to receive all AGARD publications, or just those relating to one or more specific AGARD Panels, they may be willing to include you (or your organisation) on their distribution list. AGARD publications may be purchased from the Sales Agencies listed below, in photocopy or microfiche form.

NATIONAL DISTRIBUTION CENTRES

BELGIUM
Coordonnateur AGARD — VSL
Etat-major de la Force aérienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Bruxelles

CANADA
Director Scientific Information Services
Dept of National Defence
Ottawa, Ontario K1A 0K2

DENMARK
Danish Defence Research Establishment
Ryvangs Allé 1
P.O. Box 2715
DK-2100 Copenhagen Ø

FRANCE
O.N.E.R.A. (Direction)
29 Avenue de la Division Leclerc
92322 Châtillon Cedex

GERMANY
Fachinformationszentrum
Karlsruhe
D-76344 Eggenstein-Leopoldshafen 2

GREECE
Hellenic Air Force
Air War College
Scientific and Technical Library
Dekelia Air Force Base
Dekelia, Athens TGA 1010

ICELAND
Director of Aviation
c/o Flugrad
Reykjavik

ITALY
Aeronautica Militare
Ufficio del Delegato Nazionale all'AGARD
Aeroporto Pratica di Mare
00040 Pomezia (Roma)

LUXEMBOURG
See Belgium

NETHERLANDS
Netherlands Delegation to AGARD
National Aerospace Laboratory, NLR
P.O. Box 90502
1006 BM Amsterdam

NORWAY
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25
N-2007 Kjeller

PORTUGAL
Força Aérea Portuguesa
Centro de Documentação e Informação
Alfragide
2700 Amadora

SPAIN
INTA (AGARD Publications)
Pintor Rosales 34
28008 Madrid

TURKEY
Millî Savunma Başkanliği (MSB)
ARGE Dairesi Başkanliği (MSB)
06650 Bakanliklar-Ankara

UNITED KINGDOM
Defence Research Information Centre
Kentigern House
65 Brown Street
Glasgow G2 8EX

UNITED STATES
NASA Headquarters
Code JOB-1
Washington, D.C. 20546

The United States National Distribution Centre does NOT hold stocks of AGARD publications.
Applications for copies should be made direct to the NASA Center for AeroSpace Information (CASI) at the address below.
Change of address requests should also go to CASI.

SALES AGENCIES

NASA Center for
AeroSpace Information (CASI)
800 Elkridge Landing Road
Linthicum Heights, MD 21090-2934
United States

ESA/Information Retrieval Service
European Space Agency
10, rue Mario Nikis
75015 Paris
France

The British Library
Document Supply Centre
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
United Kingdom

Requests for microfiches or photocopies of AGARD documents (including requests to CASI) should include the word 'AGARD' and the AGARD serial number (for example AGARD-AG-315). Collateral information such as title and publication date is desirable. Note that AGARD Reports and Advisory Reports should be specified as AGARD-R-nnn and AGARD-AR-nnn, respectively. Full bibliographical references and abstracts of AGARD publications are given in the following journals:

Scientific and Technical Aerospace Reports (STAR)
published by NASA Scientific and Technical
Information Division
NASA Headquarters (JTT)
Washington D.C. 20546
United States

Government Reports Announcements and Index (GRA&I)
published by the National Technical Information Service
Springfield
Virginia 22161
United States
(also available online in the NTIS Bibliographic
Database or on CD-ROM)

Printed by Canada Communication Group
45 Sacré-Cœur Blvd., Hull (Québec), Canada K1A 0S7

ISBN 92-836-1009-1